



TUGAS AKHIR - TE141599

**PENGENDALIAN LEVEL AIR PADA MODEL PLANT
YOKOGAWA TIPE S4 MENGGUNAKAN PID PREDIKTIF**

Adi Prasetyawan
NRP 2211100006

Dosen Pembimbing
Ir. Rusdhianto Effendie A.K, M.T.
Juwari Purwo Sutikno, S.T., M.Eng., Ph.D.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2015



FINAL PROJECT - TE141599

***WATER LEVEL CONTROL IN TYPE S4 PLANT MODEL
YOKOGAWA USING PREDICTIVE PID***

Adi Prasetyawan
NRP 2211100006

Supervisor
Ir. Rusdhianto Effendie A.K, M.T.
Juwari Purwo Sutikno, S.T., M.Eng., Ph.D.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015

**PENGENDALIAN LEVEL AIR PADA MODEL PLANT
YOKOGAWA TIPE S-4 MENGGUNAKAN PID PREDIKTIF**

TUGAS AKHIR

**Diajukan untuk Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada**

**Bidang Studi Teknik Sistem Pengaturan
Jurusan Teknik Elektro
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember**

Menyetujui :

Dosen Pembimbing I,

Dosen Pembimbing II,

Ir. Rusdhianto Effendie A.K., MT.

Juwari Purwo S., ST, M.Eng, Ph.D.

NIP: 195704241985021001

NIP: 197306151999031003

**SURABAYA
JUNI, 2015**

PENGENDALIAN LEVEL AIR PADA MODEL PLANT YOKOGAWA TIPE S4 MENGGUNAKAN PID PREDIKTIF

Nama : Adi Prasetyawan
Pembimbing I : Ir. Rusdhianto Effendie A.K, MT.
Pembimbing II : Juwari Purwo Sutikno, S.T, M.Eng, Ph.D

ABSTRAK

Model Plant Tipe S4 keluaran Yokogawa tahun 1982 ini merupakan miniatur lengkap pengendalian *level*, *flow*, *pressure*, dan *temperature* yang ada di industri. Plant ini terdiri dari dua tangki vertikal. Tangki pertama (T1) untuk pengendalian *temperature* dan *pressure*. Sementara pada tangki kedua (T2) untuk pengendalian *flow* dan *level*. Akan tetapi, *plant* tersebut masih menggunakan sistem pengendali analog yaitu sistem pengendali pneumatik. Oleh karena itu, pada penelitian ini *plant* mengalami modifikasi agar dapat dikendalikan menggunakan *personal computer* (PC). Dengan perubahan tersebut, *plant* mampu dikendalikan menggunakan kontroler digital yaitu PID Prediktif. Pada penelitian ini, didesain kontroler PID Prediktif untuk mempertahankan level air pada Tangki 2 meskipun terjadi perubahan parameter *plant*. Berdasarkan hasil simulasi, kontroler PID Prediktif dapat mengatasi perubahan beban yang diberikan serta mampu mempertahankan *level* sesuai dengan *set point*. Hal ini ditunjukkan dengan *error steady state* sebesar 0% pada beban nominal, minimal, dan maksimal. Sementara itu, pada implementasi, respon mampu mengikuti perubahan *set point* yang diberikan dengan *error steady state* sebesar 4,568%. Pada saat diberikan gangguan sesaat, respon mampu kembali pada *steady state* dalam waktu 90 detik. Sementara saat diberikan beban, respon mempunyai *error steady state* sebesar 3,57% pada beban nominal, 1,242% pada beban maksimal, dan 4,262% pada beban minimal.

Kata Kunci : Model PlantYokogawa Tipe S4, Pengaturan *Level*, PID Prediktif

-Halaman ini sengaja dikosongkan-

WATER LEVEL CONTROL IN TYPE S4 PLANT MODEL YOKOGAWA USING PREDICTIVE PID

Name : Adi Prasetyawan
Advisor I : Ir. Rusdhianto Effendie A.K, MT.
Advisor II : Juwari Purwo Sutikno, S.T, M.Eng, Ph.D

ABSTRACT

Model Plant Tipe S4 Yokogawa is a completely miniature control system of level, flow, pressure, and temperature in the industry. Model Plant Type S4 Yologawa is composed of two vertical tanks. The Tank 1 (T1) is to controlling temperature and pressure. Meanwhile, The Tank 2 (T2) is to controlling level and flow. However, the plant is stil using analog controller, such as pneumatic. Therefore, in this final project, plant had modified. So, Model Plant Tipe S4 Yologawa could be controlled using a personal computer (PC). With these changes, plant could be controlled using a digital controller, such as Predictive PID. In this final project, Predictive PID controlleris designed to maintain water level in the Tank 2. The controller is designed so that water level suitable with set point even though plant paramater changes. Based on the result of testing, Predictive PID can handle the given load changes and able to maintain the level response at set point. Therefore, the level can follow the changes set point. In simulation, this is indicated by 0 % steady state error at nominal load, minimal load, and maksimal load. In implementation, response can follow set point changes with 4,568% error steady state. Furthermore, when shortly disturbance is given, the response was able to return to the steady state within 90 s. Meanwhile, when response is given load, there are 3,57% at nominal load, 1,242% at maximum load, and 4,262% at minimum load.

Keywords : Model Plant Tipe S4 Yokogawa, Level Control, Predictive PID

-Halaman ini sengaja dikosongkan-

KATA PENGANTAR

Segala puji dan syukur penulis panjatkan kehadirat Allah SWT karena atas segala rahmat, taufik, dan hidayah-Nya sehingga Tugas Akhir ini dapat terselesaikan dengan baik. Shalawat dan salam juga penulis sampaikan kepada Rasulullah Muhammad SAW yang telah menjadi teladan bagi penulis.

Tugas Akhir ini disusun untuk memenuhi sebagian persyaratan guna menyelesaikan pendidikan Strata-1 pada Bidang Studi Teknik Sistem Pengaturan, Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya dengan judul:

PENGENDALIAN LEVEL AIR PADA MODEL PLANT YOKOGAWA TIPE S4 MENGGUNAKAN PID PREDIKTIF

Dalam pengerjaan Tugas Akhir ini, penulis mendapat banyak sekali bantuan dan dukungan dari berbagai pihak. Oleh karena itu, dengan segala hormat, penulis mengucapkan terima kasih kepada Bapak Ir. Rusdhianto Effendie, AK, M.T dan Juwari Purwo S, S.T, M.Eng, Ph.D, selaku dosen pembimbing yang telah sabar dalam membimbing, orang tua, seluruh keluarga, dan semua pihak yang memberikan dukungan penuh kepada penulis dalam penyelesaian Tugas Akhir ini.

Penulis menyadari bahwa pengerjaan Tugas Akhir ini masih jauh dari sempurna. Untuk itu, penulis memohon maaf atas segala kekurangan dalam Buku Tugas Akhir ini. Semoga Tugas Akhir ini dapat memberikan manfaat dalam pengembangan ilmu pengetahuan dan teknologi.

Surabaya, Juni 2015

Penulis

-Halaman ini sengaja dikosongkan-

DAFTAR ISI

HALAMAN JUDUL	iii
PERNYATAAN KEASLIAN TUGAS AKHIR	iii
HALAMAN PENGESAHAN	v
ABSTRAK	vii
ABSTRACT	ix
KATA PENGANTAR	xi
DAFTAR ISI	xiii
DAFTAR GAMBAR	xvi
DAFTAR TABEL	xxi
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Perumusan Masalah	2
1.3 Batasan Masalah	2
1.4 Tujuan	3
1.5 Sistematika Penulisan	3
1.6 Relevansi	4
BAB II TEORI PENUNJANG	5
2.1 Pengenalan Plant Model S4 Yokogawa	5
2.1.1 Pengendalian <i>Level</i>	8
2.1.2 Pengendalian <i>Flow</i>	9
2.1.3 Komponen Pengendalian Level Model S4 Yokogawa	9
2.2 Identifikasi Sistem	10
2.2.1 Identifikasi <i>Loop</i> Tertutup	11
2.2.2 Identifikasi Metode Strejc	11
2.3 Model Predictive Control	13
2.3.1 Pendekatan <i>State Space</i>	14
2.3.2 Pendekatan Model Dikrit	17
2.4 PID Prediktif	18
2.5 Rangkaian Konverter	20
2.5.1 Konverter Arus ke Tegangan (I/V)	20
2.5.2 Konverter Tegangan ke Arus (V/I)	22
2.6 <i>Pressure Sensor</i>	22
2.7 Komunikasi <i>Serial Peripheral Interface (SPI)</i>	23
2.8 Perangkat Lunak LabVIEW 2013	24
2.8.1 Front Panel	24
2.8.2 <i>Diagram Blok</i>	24

2.9	<i>Data Acquisition</i> (DAQ)	25
2.9.1.	Mikrokontroler ATmega328P	25
2.9.2	<i>Analog to Digital Converter</i> (ADC)	26
2.9.3	<i>Digital to Analog Converter</i> (DAC)	26
BAB III PERANCANGAN SISTEM		29
3.1	Gambaran Umum Sistem	29
3.2	Modifikasi Proses Yokogawa	31
3.2.1	Penambahan Dua Instrumen (<i>pressure sensor</i>)	32
3.2.2	Penambahan Panel Kontrol	33
3.2.3	Modifikasi Aliran Air dari Tangki 1 ke Tangki 2.	35
3.2.4	Penambahan Selektor	36
3.3	Perancangan Perangkat Keras	38
3.3.1	<i>Data Acquisition</i> (DAQ)	38
3.3.2	Rangkaian Konverter	45
3.4	Identifikasi Sistem	49
3.4.1	Penarikan Garis pada Metode Strejc	50
3.4.2	Validasi Model	53
3.4.3	Model <i>Loop</i> Terbuka	56
3.5	Perancangan Kontroler	58
3.6	Perancangan Perangkat Lunak	63
3.6.1	Protokol CPU M-DAQ	63
3.6.2	Simulasi	65
3.6.2	Implementasi	66
BAB IV PENGUJIAN DAN ANALISA		73
4.1	Pengujian Komponen Sistem	73
4.1.1	Pengujian <i>Pressure Sensor</i> (IY-1 dan IY-2)	73
4.1.2	Pengujian Rangkaian Konverter	75
4.1.2	Pengujian DAC	77
4.1.3	Pengujian <i>Control Valve</i> (PCV-1)	78
4.1.4	Pengujian Sensor Model Plant Yokogawa Tipe S4	79
4.2	Pengujian Simulasi	84
4.2.1	Pengujian Respon Tanpa Kontroler	85
4.2.2	Pengujian Efek Variabel Tuning (rw) Terhadap Respon <i>Level</i>	85
4.2.3	Pengujian Efek Horizon Prediksi (Np) Terhadap Respon <i>Level</i>	87
4.2.4	Pengujian Kontroler dengan Beban	88

4.2.5	Pengujian Kontroler pada Perubahan Beban	88
4.3	Pengujian Implementasi	89
4.3.1	Pengujian Kontroler terhadap Gangguan	90
4.3.2	Pengujian Kontroler terhadap Perubahan <i>Set Point</i>	92
4.3.3	Pengujian Kontroler terhadap Perubahan Beban	93
BAB V	PENUTUP	95
5.1	Kesimpulan	95
5.2	Saran	95
DAFTAR PUSTAKA	97
LAMPIRAN	99
A.	Diagram P&ID Setelah Mengalami Perrubahan	99
B.	Diagram Pengkawatan Panel Kontrol	101
C.	Program Identifikasi pada Arduino	102
D.	Algoritma PID Prediktif	117
E.	Program Simulasi	119
F.	Program Implementasi	121
RIWAYAT PENULIS	123

-Halaman ini sengaja dikosongkan-

DAFTAR TABEL

Tabel 2.1 Daftar instrumen yang digunakan pada plant Model S-4 Yokogawa	6
Tabel 2.2 Parameter Identifikasi <i>Strejc</i>	12
Tabel 2.3. Spesifikasi Maksimum IC	21
Tabel 2.4 Spesifikasi IC	22
Tabel 2.5 Fungsi Pin IC MCP3204	26
Tabel 2.6 Konfigurasi dan Fungsi Pin IC	27
Tabel 3.1 Daftar Komponen dan Instrumen yang Dibutuhkan	32
Tabel 3. 2 Spesifikasi DAQ	38
Tabel 3.3 Konfigurasi pin antar IC SN75179B	44
Tabel 3.4 Spesifikasi IV-1 dan IV-2	47
Tabel 3. 5 Model Plant <i>Loop</i> Tertutup	53
Tabel 3.6 Nilai RSSE pada Setiap Kondisi Beban	55
Tabel 3.7 Model <i>Loop</i> Tertutup Terpilih	56
Tabel 3.8 Model <i>Loop</i> Terbuka	58
Tabel 3.9 Model Diskrit Plant	58
Tabel 3.10 Variasi Gain PID Prediktif dari <i>Tuning (rw)</i>	63
Tabel 3.11 Variasi Gain PID Berdasarkan <i>Tuning (rw)</i>	63
Tabel 3.12 Paket <i>Command</i>	64
Tabel 3.13 Paket <i>Response</i>	64
Tabel 3.14 Commad ke M-DAQ	64
Tabel 3. 15 <i>Response</i> dari M-DAQ	65
Tabel 4. 1 Hasil Regresi pada <i>Level</i>	81
Tabel 4.2 Hasil Regresi pada <i>Level</i>	83
Tabel 4.3 Spesifikasi untuk Setiap Kontroler	86
Tabel 4.4 Perbandingan Spesifikasi dari Horizon Prediksi	87
Tabel 4. 5 Spesifikasi Peforma Kontroler PID Prediktif pada Berbagai Kondisi	88
Tabel 4.6 Spesifikasi Praktis Respon <i>Level</i> pada Simulasi dengan Tiga Macam Pembebanan	89
Tabel 4.7 Spesifikasi Praktis Respon <i>Level</i> pada Implementasi dengan Tiga Macam Pembebanan	94

-Halaman ini sengaja dikosongkan-

DAFTAR GAMBAR

Gambar 2.1	Diagram P&ID Sistem Kendali Analog Plant Model S-4..	5
Gambar 2.2	Panel Kontrol Plant Model S-4 Yokogawa	8
Gambar 2.3	Mekanisme Pengendalian Level.....	8
Gambar 2.4	<i>Control Valve</i> (a) dan <i>I/P Converter</i> (b)	9
Gambar 2.5	Sensor <i>Level</i> (LT-1).....	10
Gambar 2.6	<i>Flow Recorder</i> (FR-1) dan <i>Level Recorder</i> (LR-1).....	10
Gambar 2.7	Identifikasi Loop Tertutup	11
Gambar 2.8	Penarikan Garis pada Metode Strejc.....	11
Gambar 2.9	Konsep Model Predictive Control	14
Gambar 2.10	Struktur kontroler MPC [3]	17
Gambar 2.11	Struktur PID Prediktif	20
Gambar 2.12	Konfigurasi Pin RCV420	21
Gambar 2.13	Aplikasi IC Untuk Konverter 4-20 mA ke 0-5 V	21
Gambar 2.14	Hubungan Masukan dan Keluaran Sensor	22
Gambar 2.15	Pin PSAN-01C(P)A	23
Gambar 2.16	Konfigurasi Komunikasi SPI.....	23
Gambar 2.17	Tampilan Front Panel Program Labview	24
Gambar 2. 18	Tampilan Blok Diagram LabVIEW.....	25
Gambar 2.19	Konfigurasi Pin ATmega328P	25
Gambar 2.20	Konfigurasi Pin MCP3204	26
Gambar 3.1	Skema Perubahan Pengendalian Plant	29
Gambar 3.2	Skema Perancangan Sistem.....	30
Gambar 3.3	Diagram Alir Perancangan Modifikasi Plant Yokogawa S4	30
Gambar 3.4	Hasil Perubahan Plant Model S4 Yokogawa	31
Gambar 3.5	Mekanisme Penyesuaian Sinyal Pneumatik.....	33
Gambar 3.6	Konfigurasi Tata Letak Komponen di Panel Kontrol	33
Gambar 3.7	Realisasi Panel Kontrol	35
Gambar 3.8	Modifikasi Jalur Flow Air	36
Gambar 3.9	Skema Jalur Selektor.....	36
Gambar 3.10	Terminal Y2.....	37
Gambar 3.11	Modifikasi <i>Wiring</i> Terminal Y2	37
Gambar 3.12	Realisasi DAQ	39
Gambar 3. 13	Minimum Sistem CPU DAQ.....	39
Gambar 3. 14	<i>Digital Input</i> CPU.....	40
Gambar 3. 15	<i>Digital Output</i> DAQ.....	41
Gambar 3.16	Konfigurasi Pin Slave Bus	41

Gambar 3. 17 Rangkaian Analog to Digital Converter (ADC)	42
Gambar 3.18 Rangkaian Digital to Analog Converter (DAC)	43
Gambar 3.19 Skematik IC SN75179B	44
Gambar 3. 20 Konfigurasi Modul Komunikasi dari DAQ ke Komputer	44
Gambar 3.21 Realisasi Modul Komunikasi RS485	45
Gambar 3.22 Skematik Rangkaian RCV420	46
Gambar 3.23 Rangkaian Proteksi 1	46
Gambar 3.24 Rangkaian Proteksi 2	47
Gambar 3.25 Realisasi IV-1 dan IV-2	47
Gambar 3. 26. Rangkaian Skematik VI	48
Gambar 3.27 Realisasi VI-1	48
Gambar 3.28 Diagram Blok Identifikasi	49
Gambar 3.29 Front Panel Identifikasi Plant dengan NI LabVIEW 2013	49
Gambar 3. 30 <i>Needle Valve</i> (NV-2)	50
Gambar 3.31 Penarikan Garis <i>Strejc</i> pada Kurva Respon Beban Nominal	51
Gambar 3.32 Penarikan Garis <i>Strejc</i> pada Kurva Respon Beban Minimal	52
Gambar 3.33 Penarikan Garis <i>Strejc</i> pada Kurva Respon Beban Maksimal	52
Gambar 3.34 Validasi Model dengan <i>Plant</i> untuk Beban Nominal	54
Gambar 3.35 Validasi Model dengan <i>Plant</i> untuk Beban Minimal	54
Gambar 3.36 Validasi Model dengan <i>Plant</i> untuk Beban Maksimal ..	55
Gambar 3.37 Diagram Blok Identifikasi	56
Gambar 3.38 Struktur Kontroler PID Prediktif	61
Gambar 3.39 Algoritma PID Prediktif di <i>Mathscript</i>	65
Gambar 3.40 Diagram Blok Simulasi	66
Gambar 3.41 <i>SubVI</i> Membaca Nilai ADC	66
Gambar 3.42 Tampilan Front Panel <i>readADC.vi</i>	67
Gambar 3.43 Blok Diagram <i>readADC.vi</i>	67
Gambar 3.44 <i>SubVI</i> Menulis Nilai pada DAC	68
Gambar 3.45 Tampilan Front Panel <i>wrireDAC.vi</i>	68
Gambar 3.46 Diagram Blok <i>writeDAC.vi</i>	69
Gambar 3.47 <i>SubVI</i> Inisiasi Program	69
Gambar 3.48 Tampilan <i>Front Panel</i> Inisiasi Program	70
Gambar 3.49 Blok Diagram untuk Inisiasi Program	70
Gambar 3.50 Menutup Komunikasi	70

Gambar 3.51 Diagram Blok Implementasi	71
Gambar 4.1 Fungsi Arus terhadap Pressure IY-1	74
Gambar 4.2 Fungsi Arus terhadap Pressure IY-2	74
Gambar 4.3 Pin Konfigurasi Pengujian	75
Gambar 4.4 Fungsi Tegangan terhadap Arus IV-1	75
Gambar 4.5 Fungsi Tegangan terhadap Arus IV-2	76
Gambar 4.6 Konfigurasi Pengujian VI-1	77
Gambar 4.7 Fungsi Arus terhadap Tegangan VI-1	77
Gambar 4.8 Fungsi Bit DAC terhadap Tegangan Output	78
Gambar 4.9 Fungsi Buka-an Valve terhadap Bit DAC	78
Gambar 4. 10 Hasil Bacaan Level LT-1	80
Gambar 4.11 Regresi Bacaan Level.....	80
Gambar 4.12 Algoritma LabVIEW untuk Regresi Linier	81
Gambar 4.13 Hasil Bacaan Flow FT-1	82
Gambar 4.14 Regresi Bacaan <i>Flow</i>	83
Gambar 4.15 Algoritma LabVIEW untuk Regresi Akar	84
Gambar 4. 16 <i>Front Panel</i> Simulasi	84
Gambar 4. 17 Respon Level Tanpa Kontroler pada Beban Nominal ..	85
Gambar 4.18 Perbandingan Respon dengan Berbagai Kontroler	85
Gambar 4.19 Perbandingan Sinyal Kontrol $r_w=50$, $r_w=100$, dan $r_w=500$	86
Gambar 4.20 Perbandingan Respon Level dengan $N_p=50$, $N_p=200$, dan $N_p=1000$	87
Gambar 4.21 Respon <i>Level</i> dengan Berbagai Transisi Beban pada Simulasi	89
Gambar 4.22 Respon <i>Level</i> dengan Gangguan.....	90
Gambar 4.23 Respon Transien <i>Level</i>	91
Gambar 4.24 Pemberian Gangguan Sesaat pada <i>Valve F(L)CV-1</i>	91
Gambar 4.25 Sinyal Kontrol Akibat Pemberian Gangguan	91
Gambar 4.26 Respon <i>Level</i> dengan Perubahan <i>Set Point</i>	92
Gambar 4.27 <i>Overshoot</i> Akibat <i>Pressure Regulator</i> Naik	92
Gambar 4.28 Respon <i>Level</i> dengan Berbagai Transisi Beban pada Implementasi	93
Gambar 4.29 Perubahan Beban Mengakibatkan Respon Level Berubah	93

-Halaman ini sengaja dikosongkan-

BAB I

PENDAHULUAN

Pada bab ini, berisi tentang berbagai uraian penjelasan awal mengenai penelitian. Penjelasan tersebut meliputi latar belakang, perumusan masalah, batasan masalah, tujuan, dan relevansi.

1.1 Latar Belakang

Model Plant Tipe S4 keluaran Yokogawa tahun 1982 ini merupakan miniatur dari otomasi sistem kendali *level*, *flow*, *pressure*, dan *temperature* yang ada di industri. Masing-masing proses berjalan secara independen, tidak terikat satu sama lain. Dalam hal ini, Model Plant Yokogawa Tipe S4 merupakan jenis *plant single tank*. *Plant* ini terdiri dari dua tangki vertikal.

Tangki pertama untuk pengendalian *temperature* dan *pressure*. Sementara pada tangki kedua untuk pengendalian *flow* dan *level*. Akan tetapi semua instrumen yang digunakan masih berjenis analog. Hal ini dapat diamati pada sinyal-sinyal yang ditransmisikan antar instrumen, yaitu berupa sinyal pneumatik dan sinyal listrik (tegangan dan arus).

Pada Tangki Pertama (T1) menggunakan kontrol elektrik sedangkan pada Tangki Kedua (T2) menggunakan kontrol pneumatik. Selain itu, metode kontrol yang digunakan pada *plant* tersebut cukup sederhana, yaitu kontroler proporsional (proses pengendalian *level* dan *flow*), kontroler proporsional-integral (proses pengendalian *pressure*), dan ON-OFF kontrol (proses pengendalian *temperature*) pada [1].

Oleh karena itu, sebelum merancang kontroler, pada Tugas Akhir ini akan dirancang sistem kendali digital berbasis *personal computer* (PC) untuk Model Plant Yokogawa Tipe S4. Proses digitalisasi pada *plant* ini bertujuan untuk menerapkan metode kontrol digital (*PC Based*) yaitu PID Prediktif. Melalui proses digitalisasi tersebut, kontroler PID Prediktif dapat diimplementasikan pada *riil plant*.

Tujuan dari PID Prediktif dalam pengaturan *level* adalah untuk mengendalikan *level* agar sesuai dengan *set point* yang diinginkan meskipun terjadi perubahan beban dan terdapat *time delay*. Pada pengaturan *level*, perubahan beban terjadi akibat *flow* keluar yang berbeda-beda. Varisi beban dilakukan dengan cara memvariasi besarnya putaran katup beban (NV-2). Pada Tugas Akhir ini yang akan dibahas adalah pengendalian *level* pada Tangki 2. Prinsip PID Prediktif adalah

memprediksi hasil keluaran selanjutnya. Hasil keluaran selanjutnya dapat ditentukan dari data keluaran referensi sekarang dan masukan sebelumnya sehingga menghasilkan keluaran yang akan datang menggunakan pendekatan model diskrit seperti pada [2].

Perhitungan *gain* dari PID prediktif tersebut menggunakan *Model Predictive Control* (MPC) seperti pada [3]. PID dibuat sedemikian rupa sehingga mempunyai kemampuan prediktif seperti pada [4]. Untuk melihat hasil perancangan kontroler, metode kontrol yang diajukan akan diuji pada Model Plant Yokogawa Tipe S4.

1.2 Perumusan Masalah

Dalam Tugas Akhir ini, permasalahan yang akan diselesaikan adalah perancangan sistem kendali berbasis *personal computer* (PC). Hal ini dikarenakan sistem kendali pada Plant Model Yokogawa Tipe S4 masih bersifat analog yaitu kendali pneumatik. Sehingga sulit untuk mengimplementasikan sistem kendali berbasis komputer seperti kontroler PID Prediktif.

Selain itu permasalahan yang akan diselesaikan adalah permasalahan yang diakibatkan oleh perubahan parameter proses karena perubahan pembebanan. Berubahnya parameter proses dapat mengakibatkan berubahnya respon *level*. Serta mengatasi permasalahan *time delay*. Mengacu pada permasalahan tersebut, maka diperlukan suatu metode sistem kendali untuk menyelesaikan masalah tersebut, sehingga pada Tugas Akhir ini metode kontrol yang digunakan adalah PID Prediktif.

1.3 Batasan Masalah

Dalam Tugas Akhir ini, permasalahan dibatasi pada *plant* yang digunakan yaitu *Plant Model S-4 Tipe Yokogawa* pada proses di Tangki 2. Kontroler didesain pada kondisi beban nominal. Kontroler yang sama diuji pada beban minimal dan maksimal. *Set point* (masukan) *level* tidak lebih dari 80 % dari *full range* kerja sistem karena ketidaklinieran sistem. Dalam implementasi kondisi operasi *set point* dipilih 50%. Untuk parameter kontroler PID, *gain* dari masing-masing parameter yang diuji adalah hasil dari perhitungan *gain* PID Prediktif. Yang dipisahkan unsur PID nya.

1.4 Tujuan

Tujuan dari penelitian ini adalah untuk membuat mode sistem kendali digital berbasis PC (*personal computer*) pada plant Model S-4 Yokogawa tanpa menghilangkan/mengganti kontroler sistem sebelumnya yang berupa berupa analog. Sehingga *plant* ini memiliki 2 pilihan mode kendali yaitu analog dan digital berbasis PC. Diharapkan, melalui sistem kendali yang berbasis PC ini mampu mengimplementasikan berbagai kontroler yang baru.

Selain itu, pada penelitian ini akan dirancang kontroler PID Prediktif berbasis *Model Predictive Control* yang akan diimplementasikan pada Plant Model S4 Yokogawa. Hasil perancangan kontroler ini berupa *gain* kontroler PID sehingga diharapkan mampu mempertahankan *level* air sesuai dengan *set point* yang diharapkan. Adapun hasil dari penelitian ini diharapkan dapat memberikan kontribusi kepada laboratorium Teknik Kimia dalam praktikum *process control*. Hal ini dikarenakan, Plant Model S4 Yokogawa sudah berbasis digital sehingga dapat diimplementasikan berbagai kontroler digital lainnya.

1.5 Sistematika Penulisan

Pembahasan Tugas Akhir ini akan dibagi menjadi lima Bab dengan sistematika sebagai berikut:

BAB I : PENDAHULUAN

Bab ini meliputi latar belakang, perumusan masalah, batasan masalah, tujuan penelitian, sistematika laporan, dan relevansi.

BAB II : TEORI PENUNJANG

Bab ini menjelaskan tentang tinjauan pustaka, penjelasan umum Plant Model Yokogawa Tipe S4, Identifikasi plant melalui metode Sterj, *Model Predictive Control* (MPC), dan PID Prediktif. Selain itu, terdapat pula berbagai teori dasar dari perancangan *hardware* yaitu *signal conditioning* (I/V dan V/I) dan *Data Acquisition* (DAQ).

BAB III : PERANCANGAN SISTEM

Bab ini membahas desain dan perancangan algoritma kontrol untuk metode kontrol berdasarkan teori penunjang pada Bab II. Serta membahas perancangan digitalisasi Model Plant Yokogawa Tipe S4.

BAB IV : HASIL PENGUJIAN DAN ANALISA

Bab ini memuat hasil simulasi dan hasil implementasi serta analisis dari hasil tersebut.

BAB V : PENUTUP

Bab ini berisi kesimpulan dan saran dari hasil pengujian yang telah diperoleh.

1.6 Relevansi

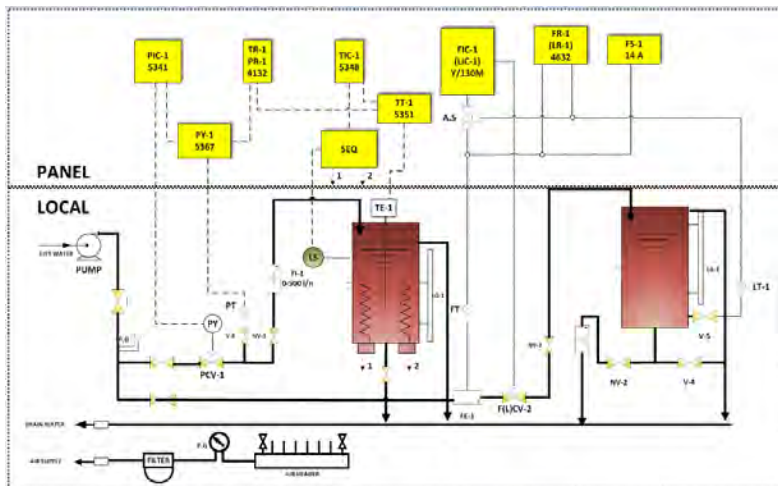
Hasil yang diperoleh dari Tugas Akhir ini diharapkan menjadi referensi perancangan desain kontroler PID Prediktif untuk pengendalian proses. Selain itu, hasil dari perancangan digitalisasi *plant* ini diharapkan mampu dipakai sebagai alat praktikum di Laboratorium Teknik Kimia ITS.

BAB II TEORI PENUNJANG

Pada bab ini, berisi tentang teori penunjang dari berbagai pustaka yang mendukung Tugas Akhir. Uraian dari tiap-tiap pustaka yang disampaikan akan dikerucutkan untuk mendapatkan beberapa konsep yang digunakan dalam teori penunjang penelitian. Teori penunjang yang dimaksud diantaranya adalah pengenalan Plant Model S-4 Yokogawa, identifikasi plant, *Model Predictive Control* (MPC), PID Prediktif, rangkaian konverter, *pressure sensor*, komunikasi SPI, NI LabVIEW 2013, dan *Data Acquisition* (DAQ).

2.1 Pengenalan Plant Model S4 Yokogawa [1]

Pada Plant Model S-4 Yokogawa ini terdapat 4 proses yang dikendalikan, yaitu *level*, *temperature*, *flow*, dan *pressure*. Masing-masing proses berjalan secara independen, tidak terikat satu sama lain. Hal ini akan nampak jelas pada Gambar 2.1.



Gambar 2.1 Diagram P&ID Sistem Kendali Analog Plant Model S-4 Yokogawa

Pada jalur pipa, dari *valve* V-1 hingga mencapai Tangki T1 terdapat dua proses pengendalian, yaitu *pressure* dan *temperature*. Pengendalian *pressure* dilakukan pada cairan dalam pipa yang menuju Tangki T1. Sedangkan pengendalian *temperature* dilakukan pada cairan yang terdapat dalam tangki. Pada tangki T1, sinyal yang diolah berupa sinyal listrik berupa besaran arus dan tegangan.

Sedangkan, pada jalur pipa dari *valve* V-2 hingga mencapai tangki T2 terdapat dua proses pengendalian pula, yaitu *flow* dan *level*. Pengendalian *flow* dilakukan pada cairan yang lewat dalam pipa menuju tangki T2. Sedangkan pengendalian level dilakukan pada cairan yang berada di dalam tangki T2. Seluruh proses ini dikendalikan melalui perangkat yang terdapat pada panel kontrol. Pada Tabel 2.1 terdapat instrumen yang digunakan keempat proses pada Plant Model S-4 Yokogawa.

Tabel 2.1 Daftar instrumen yang digunakan pada plant Model S-4 Yokogawa

No	Proses	Instrumen	Peran	Sinyal Transmisi	Besaran
1.	Pengen- dalian <i>Pressure</i> (proses kontinu)	PIC-1	Kontroler	Input : tegangan listrik Output : arus listrik	1-5 Volt 4-20 mA
		PCV-1	Aktuator	Input : arus listrik Output : bukaan valve	4-20 mA %opening
		PT-1	Sensor	Input : <i>pressure</i> Output : arus listrik	4-20 mA
		PR-1	Recorder	Input : tegangan listrik	1-5 Volt
		PY-1	Konverter	Input : arus listrik Output : tegangan listrik	4-20 mA 1-5 Volt
2.	Pengen- dalian <i>Tempera- ture</i> (proses diskrit)	TIC-1	Kontroler	Input : tegangan listrik Output : digital	1-5 Volt High / Low
		H-1	Aktuator	Input : tegangan listrik Output : kalor	220 Volt AC 6000 Watt
		H-2	Aktuator	Input : tegangan listrik Output : kalor	220 Volt AC 6000 Watt
		TE-1	Sensor	Input : kalor Output : resistansi listrik	Ohm
		TT-1	Transmitter	Input : resistansi listrik Output : tegangan listrik	Ohm 1-5 Volt

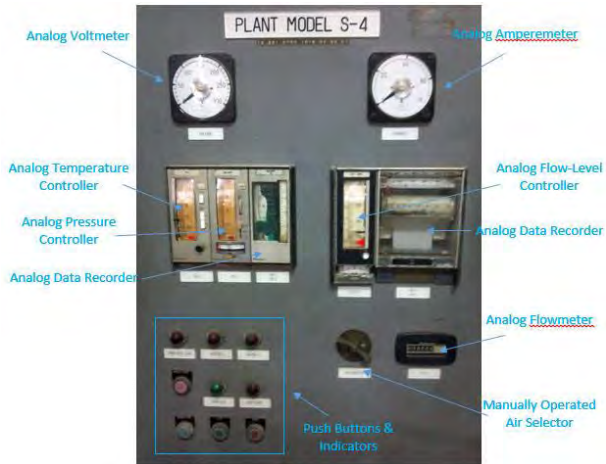
No	Proses	Instrument	Peran	Sinyal Transmisi	Besaran
	Pengendalian <i>Temperature</i> (proses diskrit)	TR-1	Recorder	Input : tegangan listrik	1-5 Volt
		SEQ	Driver	Input : digital Output : digital	High / Low High / Low
		LS	Sensor	Input : level air Output : digital	High / Low
3.	Pengendalian <i>Flow</i> (proses kontinu)	FIC-1	Kontroler	Input : pneumatik Output : pneumatik	3-15 psi 3-15 psi
		F(L)CV-1	Aktuator	Input : pneumatik Output : bukaan valve	3-15 psi %opening
		FT-1	Sensor	Input : flow rate Output : pneumatik	0-500 L/h 3-15 psi
		A.S	Air Selector		
		FR-1	Recorder	Input : pneumatik	3-15 psi
		FS-1	Switch		
4.	Pengendalian <i>Level</i> (proses kontinu)	LIC-1	Kontroler	Input : pneumatik Output : pneumatik	3-15 psi 3-15 psi
		F(L)CV-1	Aktuator	Input : pneumatik Output : bukaan valve	3-15 psi %opening
		LT-1	Sensor	Input : level air Output : pneumatik	0-500mm 3-15 psi
		A.S	Air Selector		
		LR-1	Recorder	Input : pneumatik	3-15 psi

Pada Tabel 2.1 diatas, terlihat bahwa instrumen yang digunakan adalah berjenis analog. Hal ini dapat dilihat pada sinyal-sinyal yang ditransmisikan antar instrumen, yaitu berupa:

- 1.Sinyal pneumatik
- 2.Sinyal listrik – Tegangan
- 3.Sinyal listrik – Arus

Metode kontrol yang digunakan pada *plant* tersebut cukup sederhana, yaitu kontroler proporsional pada proses pengendalian *level* dan *flow*, kontroler proporsional-integral pada proses pengendalian *pressure*, dan kontrol ON-OFF pada proses pengendalian *temperature*. Hal ini juga nampak pada instrumen kontrol pada *plant* yang masih

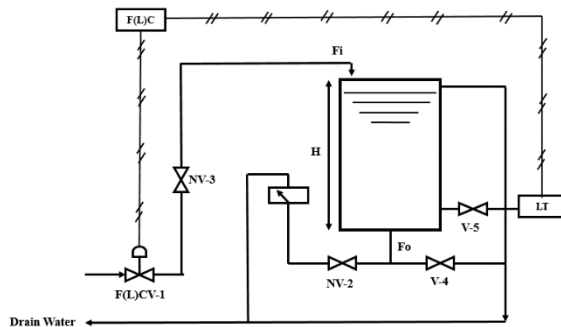
menggunakan komponen-komponen analog, untuk lebih jelasnya lihat Gambar 2.2.



Gambar 2.2 Panel Kontrol Plant Model S-4 Yokogawa

2.1.1 Pengendalian *Level*

Plant model S-4 mendemonstrasikan pengendalian *level* dalam sebuah tangki tegak berpenampang lingkaran. *Flowrate* air masuk dipakai sebagai variabel pengendali *level* dalam tangki dengan sebuah katup pengendali pneumatik sebagai aktuatur. *Flowrate* keluar sebagai beban untuk lebih detail mengenai mekanisme pengendalian *level*, dapat dilihat pada Gambar 2.3.



Gambar 2.3 Mekanisme Pengendalian Level

2.1.2 Pengendalian Flow

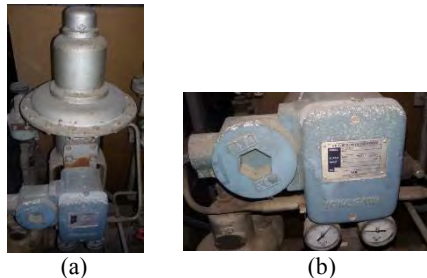
Plant model S-4 mendemonstrasikan pengendalian aliran dalam sebuah pipa pendek. *Pressure drop* pada katup pengendali dipakai sebagai variabel. Tekanan pompa maupun bukaan kran disepanjang pipa (kecuali katup pengendali) adalah sumber gangguan pada sistem ini. Pada *plant*, pengendalian *flow* menggunakan kontroler proporsional.

2.1.3 Komponen Pengendalian Level Model S4 Yokogawa

2.1.3.1 Control Valve (PCV-1)

Control valve seperti pada Gambar 2.4 dioperasikan oleh pengubah arus ke pneumatik (*I/P Converter*). Pengubah arus ke pneumatik pada Gambar 2.4 menerima sinyal kontrol sebesar 4-20 mA dan mengubahnya menjadi sinyal *pneumatik* sebesar 3-15 psi. Sementara untuk *supply*-nya 1,4 - 1,8 kg/cm².

Control valve memiliki karakteristik *air to open*, di mana katup membuka jika diberikan tekanan, bila sinyal kontrol yang diberikan maksimal maka katup kontrol pneumatik akan membuka sempurna. Bukaan maksimum (100 %) dari PCV-1 adalah 1 cm.



Gambar 2.4 *Control Valve* (a) dan *I/P Converter* (b)

2.1.3.2 Sensor Level (LT-1)

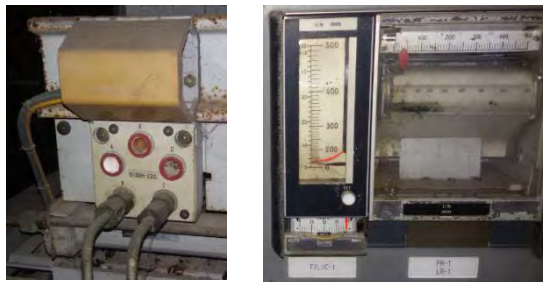
Berbeda dengan sensor pada umumnya, *sensor level* (LT-1) seperti pada Gambar 2.5 keluarannya tidak berupa sinyal listrik namun sinyal pneumatik 3-15 psi. Sementara untuk masukannya berupa ketinggian level air dari 0-500 mm. Komponen ini mempunyai *supply* tekanan 3-15 psi



Gambar 2.5 Sensor *Level* (LT-1)

2.1.3.3 *Flow Recorder* (FR-1) dan *Level Recorder* (LR-1)

Baik FR-1 maupun LR-1 seperti pada Gambar 2.6 merupakan komponen untuk merekam nilai hasil bacaan FT-1 dan LT-1. Besaran yang diolah berupa sinyal pneumatik 3-15 psi. Hasil rekaman akan dicetak di kertas yang terletak pada panel. Dengan adanya FR-1/LR-1, nilai besaran terukur dapat dilihat pada panel.



Gambar 2.6 *Flow Recorder* (FR-1) dan *Level Recorder* (LR-1)

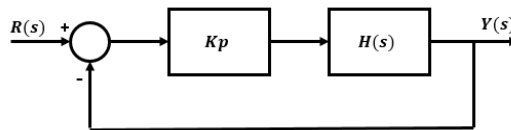
2.2 Identifikasi Sistem [5] [6]

Identifikasi sistem adalah proses mendapatkan model matematis suatu *plant* atau sistem berdasarkan hubungan masukan dan keluaran dari data hasil pengukuran. Identifikasi dapat dilakukan secara *loop* tertutup maupun *loop* terbuka. Pada bagian ini hanya dijelaskan identifikasi *loop* tertutup.

2.2.1 Identifikasi Loop Tertutup

Identifikasi ini dilakukan dengan memberikan sinyal masukan pada sistem pengaturan proses dengan aksi penguatan loop tertutup untuk mendapatkan fungsi penghantar model.

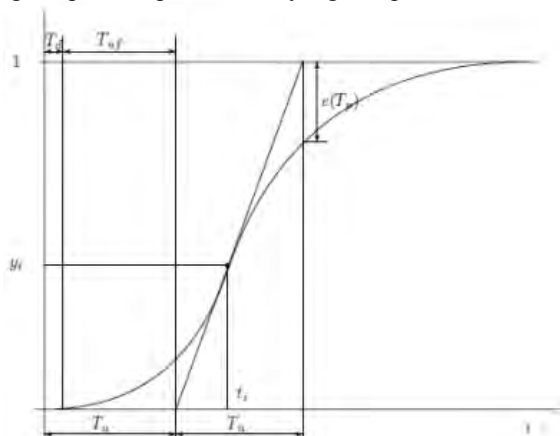
Penggunaan kontrol *proporsional* pada identifikasi *loop* tertutup membantu mendapatkan model matematika plant untuk model *loop* terbuka. Diagram blok sistem pengaturan loop tertutup dapat dilihat pada Gambar 2.7 .



Gambar 2.7 Identifikasi Loop Tertutup

2.2.2 Identifikasi Metode Strejc

Identifikasi dengan menggunakan metode *Strejc* dapat diterapkan terhadap plant orde tinggi yang memiliki respon *overdamped* ($\zeta > 1$). Dimana respon sistem memiliki waktu tunda (*time delay*) sebelum mencapai pada keadaan tunaknya. Pada metode Strejc, dilakukan penarikan garis pada respons sistem yang ada pada Gambar 2.8.



Gambar 2.8 Penarikan Garis pada Metode Strejc

Untuk menentukan fungsi alih *plant*/sistem melalui identifikasi *Strejc*, maka perlu dilakukan beberapa prosedur terhadap respon *plant*/sistem dalam domain waktu yang telah didapat. Prosedur tersebut sangat berkaitan dengan tabel *Strejc* seperti yang diberikan pada Tabel 2.2.

Tabel 2.2 Parameter Identifikasi *Strejc*

n	T_a/T	T_u/T	T_u/T_a
1	1,000	0,000	0,000
2	2,718	0,282	0,104
3	3,695	0,805	0,218
4	4,463	1,425	0,319
5	5,119	2,100	0,410

Mengacu pada respon step *plant*/sistem yang ditunjukkan pada Gambar 2.8, dapat dipilih fungsi penghantar *plant* pada Persamaan (2.1) :

$$G(s) = \frac{Ke^{rs}}{(1 + Ts)^2} \quad (2.1)$$

dengan parameter-parameter pada Persamaan (2.1) dicari berdasarkan prosedur metode *Strejc* :

1. Hitung nilai K , di mana K merupakan *Gain Overall*
2. Tentukan titik belok pada grafik respon step
3. Tentukan garis singgung pada titik belok
4. Dapatkan nilai T_u dan T_a
5. Dapatkan nilai n , yaitu nilai n terbesar pada tabel ketika nilai T_u/T_a lebih besar dari T_u/T pada tabel
6. Dapatkan nilai T , dengan menggunakan nilai T_u/T pada tabel
7. Dapatkan nilai T_u' , dengan $T_u' = \left(\frac{T_u}{T_a}\right)_{tabel} \cdot T_a$
8. Dapatkan nilai τ , dengan persamaan $\tau = T_u - T_u'$
9. Jika τ bernilai negatif, maka n salah, kembali ke langkah 5
10. Subtitusikan nilai K , τ , dan T pada Persamaan (2.1)

Perancangan kontroler PID Prediktif menggunakan model dalam representasi waktu diskrit seperti pada Persamaan (2.2). Oleh karena itu, pada Tugas Akhir ini akan disusun model matematis menggunakan waktu diskrit. Karena model matematis yang dihasilkan dalam bentuk waktu kontinu maka model tersebut harus diubah dahulu ke dalam waktu diskrit seperti pada (2.2). Waktu *sampling* (t_s) dari *plant* harus lebih cepat dari pada *time rise* (t_r).

$$\frac{Y(z)}{U(z)} = \frac{b_0 z^n + b_1 z^{n-1} + \dots + b_n}{z^n + a_1 z^{n-1} + \dots + a_n} \quad (2.2)$$

Setelah mendapatkan model matematis dengan menggunakan metode *Strejc*, dilakukan validasi terhadap model matematis yang sudah didapat. Validasi ini dilakukan dengan membandingkan respon sistem dengan respon model matematis aktual *plant*. Hasil perbandingan ini untuk menghitung besar selisih atau kesalahan antara kedua respon tersebut. Untuk Validasi model, digunakan metode *Root Mean Square Error* (RMSE). Perhitungan RMSE dituliskan dengan Persamaan (2.3).

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_a - y_m)^2} \quad (2.3)$$

y_a : nilai keluaran aktual

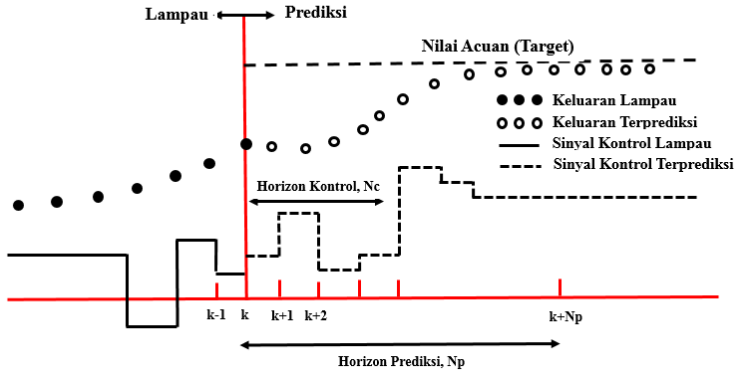
y_m : nilai keluaran pemodelan

n : jumlah data

2.3 Model Predictive Control [3]

Pada buku *Model Predictive Control System Design and Implementation Using MATLAB®* (liuping wang, 2009) menyatakan bahwa nilai yang akan datang dari variabel keluaran diprediksi menggunakan model dinamik dari suatu proses dan pengukuran saat ini. Kedua hal tersebut merupakan konsep dasar dari algoritma MPC seperti pada Gambar 2.9. Berbeda dengan metode kompensasi waktu tunda, MPC dibuat untuk lebih dari satu waktu tunda kedepan. Perhitungan aksi kontrol berdasarkan pada prediksi yang akan datang dan pengukuran saat ini. Sementara keluaran terprediksi bergantung pada nilai masukan dan keluaran lampau serta sinyal kendali yang akan datang. Variabel yang

akan dimanipulasi $u(k)$ pada *sampling* ke- k dihitung sedemikian rupa, sehingga mampu meminimalkan fungsi objektif J .



Gambar 2.9 Konsep Model Predictive Control

2.3.1 Pendekatan *State Space*

Pada MPC secara sederhana, *plant* diasumsikan sebagai *single input single output* yang dideskripsikan oleh Persamaan (2.4) berikut.

$$\left. \begin{aligned} x_m(k+1) &= \mathbf{A}_m x_m(k) + \mathbf{B}_m u(k) \\ y(k) &= \mathbf{C}_m x_m(k) \end{aligned} \right\} \quad (2.4)$$

u : variabel manipulasi/ variabel input *plant*

y : variabel output

x_m : variabel state

Perancangan kontroler MPC dilakukan dengan melalui empat tahap, yaitu membentuk *augmented model*, menghitung *predicted output* dan *future control*, melakukan optimasi terhadap error, dan mencari gain MPC dan K_y .

1. Membentuk *Augmented Model*

Augmented Model dibentuk dari Persamaan (2.4) dan Persamaan (2.5).

$$\left. \begin{aligned} \begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} &= \overbrace{\begin{bmatrix} A_m & 0_m^T \\ C_m A_m & 1 \end{bmatrix}}^A \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} + \overbrace{\begin{bmatrix} B_m \\ C_m B_m \end{bmatrix}}^B \Delta u(k) \\ y(k) &= \overbrace{\begin{bmatrix} 0_m & 1 \end{bmatrix}}^C \begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} \end{aligned} \right\} \quad (2.5)$$

Dengan, $0_m = \overbrace{[0 \ 0 \ \dots \ 0]}^{n1}$. Triplet A,B,C dinamakan *augmented* model yang akan digunakan di dalam perancangan *predictive control* pada tahap kedua.

2. Menghitung *Predicted Output* dan *Future Control*

Setelah mendapat *tripet* matriks A,B, dan C. Langkah berikutnya adalah menghitung keluaran terprediksi dan sinyal kontrol yang akan datang menggunakan Persamaan (2.6).

$$Y = Fx(k_f) + \Phi \Delta U \quad (2.6)$$

Masing-masing matriks F dan Φ mempunyai nilai seperti Persamaan (2.7)

$$F = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N_p} \end{bmatrix}; \Phi = \begin{bmatrix} CB & 0 & \dots & 0 \\ CAB & CB & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & \dots & CA^{N_p-N_c}B \end{bmatrix} \quad (2.7)$$

Untuk penjelasannya, N_p merupakan jumlah prediksi *output* yang akan dilakukan oleh *Model Predictive Control* (MPC). Sementara N_c adalah jumlah langkah kontrol berkelanjutan yang diterapkan dan diprediksi oleh pengendali *Model Predictive Control* (MPC) dalam sebuah *sampling time*. Syarat, $N_p \geq N_c$, karena perhitungan ini digunakan untuk menghitung gain PID Prediktif, maka $N_c = 3$. Sehingga, Persamaan (2.7) menjadi Persamaan (2.8)

$$F = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N_p} \end{bmatrix}; \Phi = \begin{bmatrix} CB & 0 & 0 \\ CAB & CB & 0 \\ \vdots & \vdots & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B \end{bmatrix} \quad (2.8)$$

3. Melakukan optimasi terhadap *error*

Optimasi dilakukan menggunakan indeks performansi dan turunan pertama dari indeks performansi pada Persamaan (2.9).

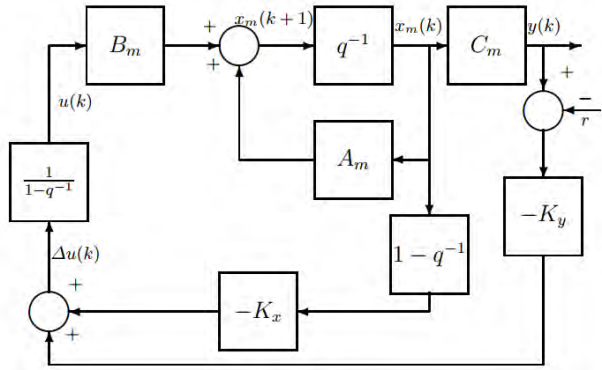
$$\left. \begin{aligned} J &= (R_s - Y)^T (R_s - Y) + \Delta U^T R \Delta U \\ \Delta U &= (\Phi^T \Phi + R)^{-1} \Phi^T (R_s - Fx(k_i)) \end{aligned} \right\} \quad (2.9)$$

Untuk sinyal referensi yang diberikan $r(k_i)$ saat waktu sampling k_i , dalam sebuah *prediction horizon*, objektif dari sistem *control horizon* adalah membawa keluaran yang terprediksi sedekat mungkin ke nilai sinyal referensi. Pada persamaan indeks performansi, persamaan pertama bertujuan untuk meminimalkan *error* antara keluaran terprediksi dengan *set point*. Sementara pada persamaan kedua merefleksikan perubahan sinyal kontrol. \bar{R} merupakan matriks diagonal dengan persamaan $\bar{R} = r_w I_{N_p \times N_p}$ ($r_w \geq 0$). r_w digunakan untuk tuning MPC. Sementara, untuk R_s merupakan data vektor yang berisikan informasi sinyal referensi seperti pada Persamaan (2.10).

$$R_s^T = \overbrace{[1 \quad 1 \quad 1 \quad \dots \quad 1]}^{N_p} r(k) \quad (2.10)$$

4. Mencari gain MPC dan K_y

Gain MPC K_{mpc} adalah baris pertama dari $\{(\Phi^T \Phi + \bar{R})^{-1} \Phi^T F\}$ sedangkan K_y adalah elemen pertama dari $\{(\Phi^T \Phi + \bar{R})^{-1} \Phi^T \bar{R}_s\}$. Dari *gain* tersebut lalu dibentuk kontrol loop tertutup dari MPC. Dengan menggunakan *augmented* design model $x(k+1) = A(k) + B \Delta u(k)$ sistem loop terbuka ditentukan dengan memasukan persamaan diatas kedalam persamaan, sehingga didapatkan $x(k+1) = Ax(k) - BK_{mpc}x(k) + BK_y r$



Gambar 2.10 Struktur kontroler MPC [3]

2.3.2 Pendekatan Model Diskrit

Secara umum hubungan antara *input* dan *output plant* direpresentasikan ke dalam bentuk model diskrit pada Persamaan (2.11) :

$$a(z)\Delta y_k = b(z)\Delta u_k \quad (2.11)$$

Prediksi menggunakan informasi dari output dan perubahan pertambahan input. Sehingga persamaan menjadi Persamaan (2.12):

$$\begin{aligned} [a(z)\Delta] y_k &= [b(z)] \Delta u_k \\ A(z)y_k &= b(z)\Delta u_k \end{aligned} \quad (2.12)$$

Dengan

$$A(z) = a(z)\Delta$$

Sehingga

$$\begin{aligned} A(z) &= 1 + A_1 z^{-1} + A_2 z^{-2} + \dots + A_n z^{-n} \\ b(z) &= b_1 z^{-1} + b_2 z^{-2} + \dots + b_m z^{-m} \end{aligned}$$

Untuk satu *sample* data, maka nilai dari ouput terprediksi menjadi Persamaan (2.13):

$$y_{k+1} + A_1 y_k + \dots + A_n y_{k-n+1} = b_1 \Delta u_k + b_2 \Delta u_{k-1} + \dots + b_m \Delta u_{k-m+1} \quad (2.13)$$

Satu langkah prediksi ke depan dapat digunakan n step prediksi menggunakan cara rekursif seperti berikut :

$$\begin{aligned} y_{k+1} + A_1 y_k + \dots + A_n y_{k-n+1} &= b_1 \Delta u_k + b_2 \Delta u_{k-1} + \dots + b_m \Delta u_{k-m+1} \\ y_{k+2} + A_1 y_{k+1} + \dots + A_n y_{k-n+2} &= b_1 \Delta u_{k+1} + b_2 \Delta u_k + \dots + b_m \Delta u_{k-m+2} \\ y_{k+3} + A_1 y_{k+2} + \dots + A_n y_{k-n+3} &= b_1 \Delta u_{k+2} + b_2 \Delta u_{k+1} + \dots + b_m \Delta u_{k-m+3} \\ y_{k+4} + A_1 y_{k+3} + \dots + A_n y_{k-n+4} &= b_1 \Delta u_{k+3} + b_2 \Delta u_{k+2} + \dots + b_m \Delta u_{k-m+4} \end{aligned}$$

Persamaan diatas dapat direpresentasikan dalam bentuk matriks:

$$C_A \begin{bmatrix} y_k \\ y_{k+1} \\ y_{k+2} \\ y_{k+3} \end{bmatrix} + H_A \begin{bmatrix} y_k \\ y_{k-1} \\ y_{k-2} \\ y_{k-3} \end{bmatrix} = C_B \begin{bmatrix} \Delta u_k \\ \Delta u_{k+1} \\ \Delta u_{k+2} \\ \Delta u_{k+3} \end{bmatrix} + H_B \begin{bmatrix} \Delta u_{k-1} \\ \Delta u_{k-2} \\ \Delta u_{k-3} \\ \Delta u_{k-4} \end{bmatrix}$$

Sehingga nilai *output* terprediksi dipengaruhi oleh *input* saat ini, input sebelumnya dan nilai *output* sebelumnya menjadi Persamaan (2.14).

$$\begin{aligned} y_{\rightarrow k+1} &= C_A^{-1} C_B \Delta u_{\rightarrow k} + C_A^{-1} H_B \Delta u_{\leftarrow k-1} + C_A^{-1} H_A y_{\leftarrow k} \\ y_{\rightarrow k+1} &= \Phi \Delta u_{\rightarrow k} + G' \Delta u_{\leftarrow k-1} + F y_{\leftarrow k} \end{aligned} \quad (2.14)$$

2.4 PID Prediktif [2] [4] [7]

PID Prediktif merupakan kontroler yang terdiri dari dua unsur, yaitu PID dan Prediktif. Untuk mendapat gain dari unsur PID dan Prediktif diturunkan dari algoritma *Model Predictive Control* (MPC). MPC akan mempunyai struktur kontroler yang sama dengan PID apabila *horizon control* sama dengan tiga ($N_c=3$). Karena MPC diturunkan dari waktu diskrit, maka PID yang digunakan merupakan kontroler dalam bentuk diskrit. Sehingga, untuk konfigurasi kontroler PID diskrit mempunyai bentuk Persamaan (2.15):

$$u(k) = K_p e(k) + K_i \sum_{n=i}^k e(n) + K_d (e(k) - e(k-1)) \quad (2.15)$$

Dengan menggunakan persamaan beda pada Persamaan (2.15), maka perubahan sinyal kontrol PID menjadi Persamaan (2.16) :

$$\Delta u(k) = K_p (e(k) - e(k-1)) + K_i e(k) + K_d (e(k) - 2e(k-1) + e(k-2)) \quad (2.16)$$

Atau dalam bentuk transformasi Z pada Persamaan (2.17):

$$u(z) = \frac{q_0 + q_1 z^{-1} + q_2 z^{-2}}{1 - z^{-1}} e(z) \quad (2.17)$$

Dengan K_p , K_i and K_d adalah gain *proportional*, *integral* dan *derivative*, sehingga sesuai dengan Persamaan (2.16) dan Persamaan (2.18) maka:

$$\left. \begin{aligned} q_0 &= K_p + K_i + K_d \\ q_1 &= -K_p - 2K_d \\ q_2 &= K_d \end{aligned} \right\} \quad (2.18)$$

Seperti diketahui pada Persamaan (2.16), perubahan sinyal kontrol dari kontroler MPC akan menjadi dasar dalam menentukan *gain* pada PID dan Prediktif. Langkah selanjutnya adalah memisahkan unsur PID dan Prediktif dari *Model Predictive Control* :

$$K_{GPC} = (\Phi^T \Phi + R)^{-1} \Phi^T \quad (2.19)$$

$$\Delta u_{mpc} = K \begin{bmatrix} 1 \\ 1 \\ \cdot \\ \cdot \\ Np \end{bmatrix} r(k) - K \begin{bmatrix} F_0 & F_1 & F_2 \end{bmatrix} \begin{pmatrix} y(k) \\ y(k-1) \\ y(k-2) \end{pmatrix} \quad (2.20)$$

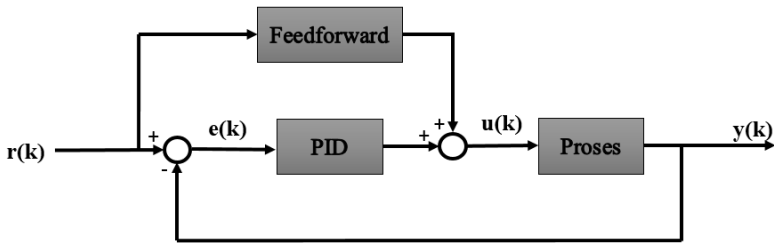
$$\Delta u_{mpc} = (Gain - feedforward) \cdot r(k) - q_0 y(k) - q_1 y(k-1) - q_2 y(k-2) \quad (2.21)$$

$$\Delta u_{mpc} = \Delta u_{feedforward} + \Delta u_{pid} \quad (2.22)$$

Melalui Persamaan (2.21) dan Persamaan (2.22) didapatkan Persamaan (2.23):

$$\left. \begin{aligned} \Delta u_{feedforward} &= (Gain - feedforward) \cdot r(k) - q_0 r(k) - q_1 r(k-1) - q_2 r(k-2) \\ \Delta u_{pid} &= q_0 e(k) + q_1 e(k-1) + q_2 e(k-2) \end{aligned} \right\} \quad (2.23)$$

Adapun struktur kontroler pada Tugas Akhir ini ditunjukkan Gambar 2.11 berikut :

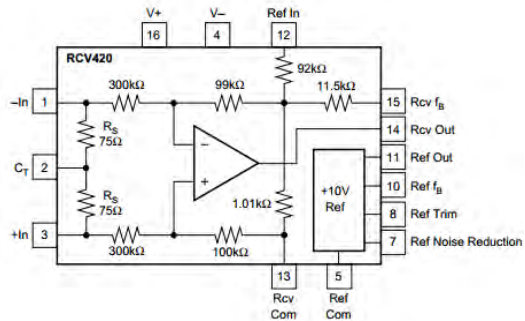


Gambar 2.11 Struktur PID Prediktif

2.5 Rangkaian Konverter

2.5.1 Konverter Arus ke Tegangan (I/V) [8]

Karena sinyal-sinyal dalam kontrol proses paling sering ditransmisikan sebagai arus, khususnya 4-20 mA, maka perlu untuk memakai sebuah konverter tegangan ke arus. Rangkaian ini harus mampu mengalirkan arus ke sejumlah beban yang berbeda tanpa mengubah karakteristik-karakteristik transfer tegangan ke arus. Pada Tugas Akhir ini, IC yang digunakan adalah RCV420 buatan pabrikan Burn-Brown. IC ini merupakan konverter arus 4-20 mA ke tegangan 0-5 V. Untuk konfigurasi pin seperti Gambar 2.12 :



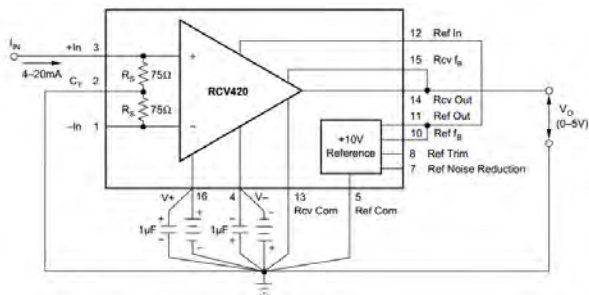
Gambar 2.12 Konfigurasi Pin RCV420

Adapun spesifikasi penggunaan maksimum besaran yang diperbolehkan pada IC disajikan pada Tabel 2.3 :

Tabel 2.3. Spesifikasi Maksimum IC

No	Besaran	Nilai maksimum
1.	Supply	$\pm 22\text{V}$
2.	Arus input, kontinu	40 mA
3.	Arus input, sesaat, 0.1 s	250 mA
4.	<i>Common-Mode Input Voltage</i> , kontinu	$\pm 22\text{V}$
5.	Suhu (<i>soldering</i> , 10s)	$\pm 300^\circ\text{C}$
6.	<i>Output Short Circuit to Common</i> (Rcv and Ref)	Kontinu

Untuk aplikasi penggunaan, IC ini hanya membutuhkan dua kapasitor $1\mu\text{F}$. Kedua kapasitor tersebut dipasang pada pin *power supply* $\pm 12\text{V}$ IC. Koneksi antar pin dari IC dijelaskan pada Gambar 2.13.



Gambar 2.13 Aplikasi IC untuk Konverter 4-20 mA ke 0-5 V

2.5.2 Konverter Tegangan ke Arus (V/I) [9]

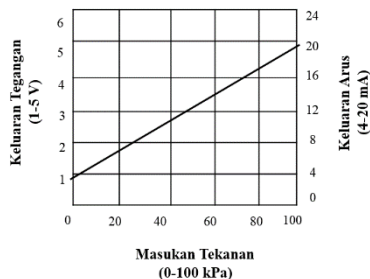
Rangkaian ini diperlukan untuk mengubah besaran tegangan 0-5 V ke 4-20 mA. Besaran tegangan yang berasal dari *Digital to Analog (D/A) converter* diubah menjadi arus. Hal ini dikarenakan masukan dari I/P merupakan besaran arus 4-20 mA. Pada Tugas Akhir ini, komponen yang digunakan adalah IC ISO-U1-P2-O1 buatan pabrikan SHENZHEN SUNYUAN TECHNOLOGY. Komponen ini merupakan jenis *isolated amplifier* untuk mengubah sinyal analog yang berupa tegangan 0-5V menjadi arus sebesar 4-20mA. IC ini memiliki tegangan kerja sebesar 12-24 VDC. Berikut adalah spesifikasi IC pada Tabel 2.4 :

Tabel 2.4 Spesifikasi IC

No	Variabel	Nilai
1.	Masukan (input)	0-5V DC
2.	Keluaran (output)	4-20 mA DC
3.	Arus input	30 mA (max)
4.	Arus output	20 mA (max)
5.	Impedansi input	1 M ohm

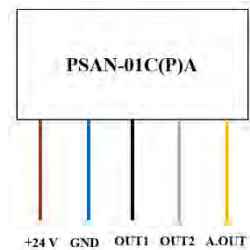
2.6 Pressure Sensor [10]

Pressure sensor ini untuk mengukur besaran tekanan pneumatik pada pipa. Besaran fisik serupa sinyal pneumatik tersebut akan diubah menjadi besaran listrik berupa arus 4-20 mA. *Digital Pressure Sensor Autonics* (PSAN Series) ini mempunyai masukan 3-14,5 psi (0-100kPa) dan keluaran 4-20 mA. Untuk hubungan masukan dan keluaran dari sensor disajikan oleh Gambar 2.14.



Gambar 2.14 Hubungan Masukan dan Keluaran Sensor

Untuk konfigurasi pin dari *pressure sensor* dapat dilihat pada Gambar 2.15:



Gambar 2.15 Pin PSAN-01C(P)A

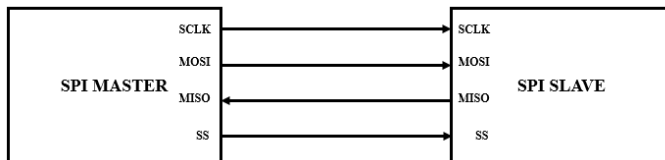
2.7 Komunikasi *Serial Peripheral Interface* (SPI) [11]

Serial Peripheral Interface (SPI) merupakan salah satu mode komunikasi serial *synchronous* kecepatan tinggi yang dimiliki oleh ATmega 328. Komunikasi SPI membutuhkan 3 jalur yaitu MOSI, MISO, dan SCK. Melalui komunikasi ini data dapat saling dikirimkan baik antara mikrokontroler maupun antara mikrokontroler dengan *peripheral* lain di luar mikrokontroler.

Komunikasi serial data antara master dan slave pada SPI diatur melalui 4 buah pin yang terdiri dari SCLK, MOSI, MISO, dan SS sebagai berikut:

- SCLK dari *master* ke *slave* yang berfungsi sebagai *clock*
- MOSI jalur data dari *master* dan masuk ke dalam *slave*
- MISO jalur data keluar dari *slave* dan masuk ke dalam *master*
- SS (*slave select*) merupakan pin yang berfungsi untuk mengaktifkan *slave*

Untuk konfigurasi antar perangkat keras dapat dilihat pada Gambar 2.16:



Gambar 2.16 Konfigurasi Komunikasi SPI

2.8 Perangkat Lunak LabVIEW 2013 [12] [13]

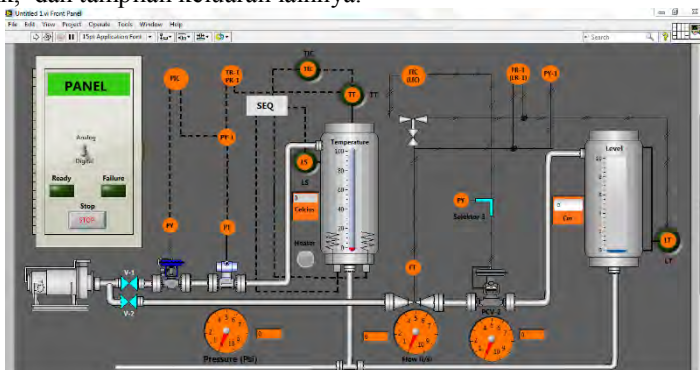
Perangkat Lunak *Labview* 2013 (*Laboratory Virtual Instrumentation Engineering Workbench*) merupakan sebuah sistem yang dikembangkan untuk instrumentasi di bidang perindustrian, eksperimen, ataupun pendidikan dan berbagai aplikasi otomasi yang yang dikembangkan berdasarkan pemrograman yang berorientasi grafis.

Labview mempunyai beberapa *toolkit* dan modul untuk analisa dan desain kontrol, pengolahan sinyal, identifikasi sistem, matematika, simulasi, dan lain-lain. *Labview* juga dilengkapi oleh fungsi-fungsi untuk analisa numerik, desain dan menggambarkan hasil data. LabVIEW 2013 mempunyai bagian dua bagian penting, yaitu *front panel* dan *block diagram*.

2.8.1 Front Panel

Front panel merupakan penghubung (*interface*) antara pengguna (*user*) dengan program aplikasi. Didalam *front panel* terdapat kontrol (*input*) dan indikator (*output*) sebagai masukan atau keluaran instrumen.

Kontrol adalah mekanisme masukan yang menyuplai data ke diagram blok yang mencakup *knop*, *push button*, *dial* dan mekanisme masukan lainnya. Sedangkan indikator adalah instrumen mekanisme keluaran yang menampilkan data dari diagram blok, mencakup grafik, , tank, dan tampilan keluaran lainnya.

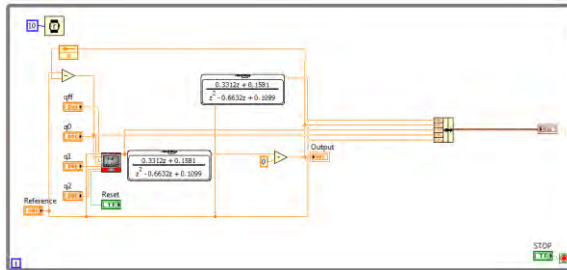


Gambar 2.17 Tampilan Front Panel Program Labview

2.8.2 Diagram Blok

Diagram blok merupakan jendela tempat menuliskan perintah dan fungsi, berisikan *source code* berupa simbol-simbol, node dan garis

sebagai data *flow* untuk mengeksekusi program termasuk kode dari *front panel*.



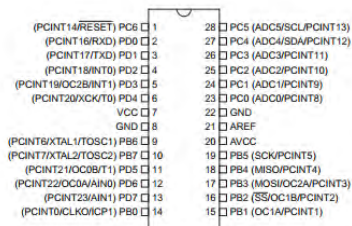
Gambar 2. 18 Tampilan Blok Diagram LabVIEW

2.9 Data Acquisition (DAQ)

Komponen DAQ ini terdiri dari tiga bagian, yaitu *Analog to Digital Converter (ADC)*, CPU, dan *Digital to Analog Converter (DAC)*. ADC bertindak sebagai analog input. Sementara DAC bertindak sebagai analog output. Untuk CPU, berfungsi untuk memproses data pada DAQ. Berikut adalah penjelasan masing-masing komponen :

2.9.1. Mikrokontroler ATmega328P [14]

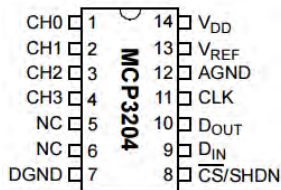
ATmega328 adalah mikrokontroler keluaran dari Atmel. ATmega328 memiliki fitur cukup lengkap, mulai dari kapasitas memori program dan memori data yang cukup besar, interupsi, *timer/counter*, PWM, USART, TWI, analog comparator, EEPROM internal dan juga ADC internal. Gambar 2.19 merupakan penjelasan mengenai konfigurasi pin-pin ATmega328P. IC tersebut digunakan didalam modul *board arduino* yang digunakan dalam penelitian dan perancangan CPU dari DAQ.



Gambar 2.19 Konfigurasi Pin ATmega328P

2.9.2 Analog to Digital Converter (ADC) [15]

Rangkaian ini bertugas untuk mengubah sinyal analog berupa tegangan 0-5 V menjadi sinyal digital yang akan diolah oleh kontroler pada komputer. Resolusi dari sinyal digital ditentukan oleh jumlah bit yang bisa direpresentasikan terhadap input analog. Untuk Tugas Akhir ini, IC ADC yang digunakan adalah MCP3204 buatan pabrik *Microchip Technology Inc* seperti pada Gambar 2.20.



Gambar 2.20 Konfigurasi Pin MCP3204

ADC ini mempunyai resolusi 12 bit dengan *sampling rate* 100 ks/s dan V_{ref} 5 V . Komponen ini juga dilengkapi 4 buah *channel analog input*. Untuk lebih jelasnya mengenai fungsi pin dapat dilihat pada Tabel 2.5.

Tabel 2.5 Fungsi Pin IC MCP3204

No	Nama Pin	Fungsi
1.	V_{DD}	+2.7V-5.5 V Power Supply
2.	DGND	Ground digital
3.	AGND	Ground analog
4.	CH0-CH7	Masukan analog
5.	CLK	Serial Clock
6.	D_{IN}	Serial Data In
7.	D_{OUT}	Serial Data Out
8.	CS	Chip Select/Shutdown Input
9.	V_{REF}	Input Tegangan Referensi

2.9.3 Digital to Analog Converter (DAC) [16]

Rangkaian ini bertugas untuk mengubah sinyal digital dari komputer menjadi besaran sinyal analog yaitu tegangan 0-5 V. Nilai dari tegangan keluaran direpresentasikan dengan Persamaan (2.24).

$$V_{out} = V_{ref} (b_1 2^{-1} + b_2 2^{-2} + b_3 2^{-3} + b_4 2^{-4} + \dots + b_n 2^{-n}) \quad (2.24)$$

Sama seperti IC ADC sebelumnya, IC DAC ini, MCP4922, juga merupakan buatan *Microchip Technology Inc.* IC ini mempunyai dua buah *channel analog out* berupa tegangan dengan V_{REF} 5 V. IC ini mempunyai tegangan keluran hingga $2 \cdot V_{REF}$ dengan mengatur *Gain Selection* (1 atau 2 kali). Selain itu, terdapat pula pin SHDN (*active low*) untuk *shutdown* kedua *channel out* dari IC. Terdapat pula pin LDAC (*active low*) untuk *update* keluaran DAC. Ketika pin ini *low*, maka kedua output V_{OUTA} dan V_{OUTB} *update* bersamaan dengan *input register*. Untuk lebih jelasnya fungsi masing-masing dari pin dapat dilihat pada Tabel 2.6.

Tabel 2.6 Konfigurasi dan Fungsi Pin IC

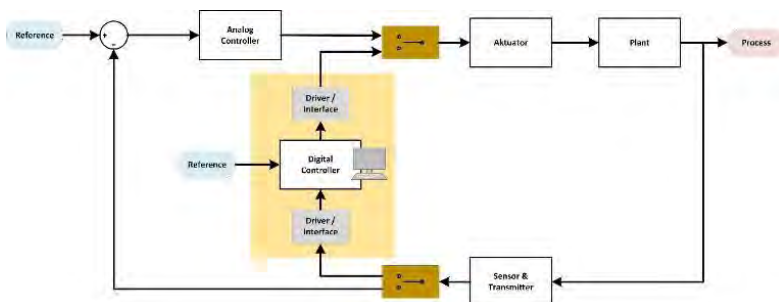
No. pin	Simbol	Fungsi
1.	V_{DD}	Tegangan masukan (+2.7V-5.5 V)
2.	NC	No Connction
3.	CS	Chip Select Input
4.	SCK	Serial Clock Input
5.	SDI	Serial Data In
6.	NC	No Connection
7.	NC	No Connection
8.	LDAC	Synchronization Input
9.	SHDN	Hardware Shutdown Input
10.	V_{OUTB}	DAC _B Out
11.	V_{REFB}	DAC _B Reference Voltage Input
12.	V_{SS}	Ground referensi untuk semua circuit
13.	V_{REFA}	DAC _A Reference Voltage Input
14.	V_{OUTA}	DAC _A Out

-Halaman ini sengaja dikosongkan-

BAB III PERANCANGAN SISTEM

3.1 Gambaran Umum Sistem

Hasil modifikasi Plant Model-S4 Yokogawa ini nantinya akan memiliki dua mode pengendalian, yaitu mode pengendali analog dan mode pengendali digital seperti pada Gambar 3.1. Pada panel lama Plant Model-S4 Yokogawa sudah terdapat sistem pengendali analog. Sedangkan, pada Tugas Akhir ini, dibuat panel kontrol baru untuk mengendalikan *plant* secara digital. Panel kontrol tersebut bertugas sebagai *interface* antara *plant* dan komputer. Terdapat sebuah selektor untuk memilih mode pengendalian yang akan diterapkan. Dalam kondisi normal, mode pengendalian yang terpilih adalah analog. Sedangkan, saat panel kontrol yang dirancang dalam kondisi aktif, maka mode pengendali yang digunakan adalah pengendali digital



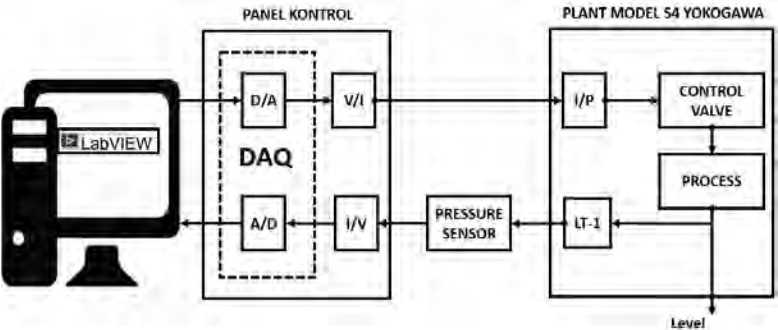
Gambar 3.1 Skema Perubahan Pengendalian Plant

Pada Tugas Akhir ini, variabel yang dikontrol adalah level air pada Tangki 2. Karena sistem pengendalian lama pada *plant* menggunakan sinyal pneumatik untuk mengendalikan, maka diperlukan instrumen tambahan untuk mengubah sinyal pneumatik tersebut menjadi sinyal listrik. Instrumen yang digunakan adalah *pressure sensor* seri PSAN buatan Pabrik Autonics. Sensor/transmitter tersebut akan mengubah keluaran LT-1 yang berupa sinyal pneumatik (3-14,5 psi) menjadi sinyal elektrik (4-20 mA).

Kemudian, dalam perancangannya, sinyal tersebut akan masuk ke *Analog to Digital Converter (A/D Converter)* pada panel kontrol. Sebelum masuk ke ADC, sinyal tersebut akan diubah dahulu menjadi

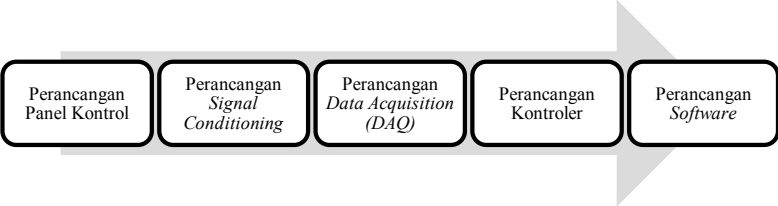
tegangan (0-5V) oleh *I/V Converter*. Setelah masuk ADC, maka sinyal tersebut akan diolah oleh kontroler PID Prediktif pada komputer dengan bantuan *software* LabVIEW 2013.

Keluaran dari kontroler tersebut berupa sinyal kontrol yang akan diolah *Digital to Analog Control (D/A Converter)*. DAC akan mengubah sinyal kontrol menjadi tegangan (0-5V). Sinyal kontrol tersebut sebelum masuk ke *I/P Converter*, akan diubah dahulu menjadi besaran arus (4-20 mA) melalui *V/I Converter*. I/P akan bertugas untuk menggerakkan *Control Valve* (PCV-1), sehingga air yang masuk ke tangki dapat dikendalikan. Untuk skema sistem pengendalian secara keseluruhan dapat dilihat pada Gambar 3.2



Gambar 3.2 Skema Perancangan Sistem

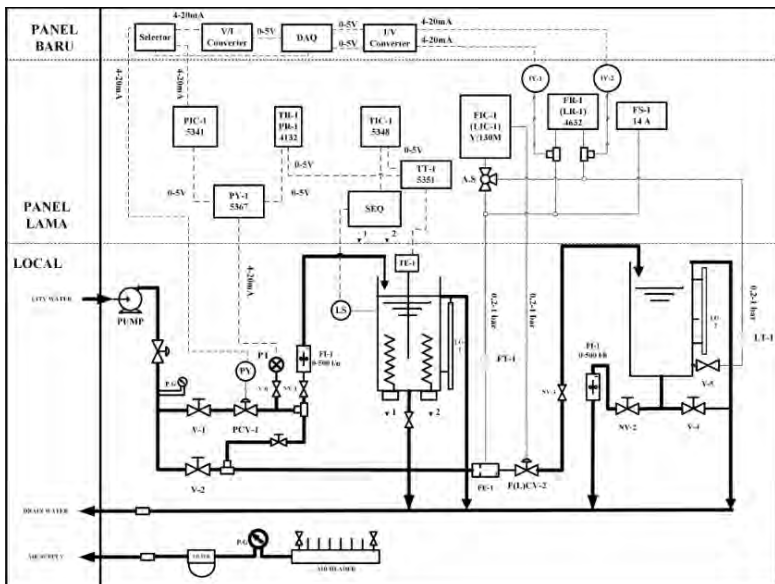
Setelah *plant* mengalami beberapa perubahan, maka hasil akhir dari modifikasi dapat dilihat pada Lampiran A dalam bentuk diagram P&ID. Secara umum untuk skema perancangan Tugas Akhir terdapat beberapa tahap yang dilalui, kesemuanya dapat dilihat pada diagram alir Gambar 3.3 berikut ini. Untuk perancangan panel kontrol, *signal conditioning*, dan *Data Acquisition (DAQ)* masuk ke dalam perancangan perangkat keras pada Subbab 3.2.



Gambar 3.3 Diagram Alir Perancangan Modifikasi Plant Yokogawa S4

3.2 Modifikasi Proses Yokogawa

Agar dapat dikendalikan menggunakan komputer, maka *plant* Yokogawa mengalami beberapa modifikasi seperti Gambar 3.4 atau pada Lampiran A. Yaitu, penambahan dua instrumen (*pressure sensor*), penambahan panel kontrol, modifikasi jalur aliran air dari tangki T1 ke tangki T2, dan penambahan selektor untuk mengubah mode pengendalian.



Gambar 3.4 Hasil Perubahan Plant Model S4 Yokogawa

Dari modifikasi yang telah dilakukan seperti pada Gambar 3.4, terdapat beberapa tambahan komponen elektronik maupun instrumen untuk mendukung proses digitalisasi. Komponen tersebut sebagian besar terletak pada panel kontrol baru. Hanya IY-1 dan IY-2 yang terletak di *local* atau *plant*. IY-1 dan IY-2 sengaja diletakkan dekat *plant* karena yang disadap adalah sinyal pneumatik. Untuk menghindari *delay* akibat panjangnya pipa yang masuk ke instrumen tersebut. Komponen yang dibutuhkan untuk digitalisasi *plant* secara detail dijelaskan pada Tabel 3.1

Tabel 3.1 Daftar Komponen dan Instrumen yang Dibutuhkan

No	Komponen	Fungsi	Sinyal
1.	IY-1 (PSAN-1)	Konverter sinyal pneumatik ke sinyal listrik	Input : 0,2-1 bar Output:4-20 mA
2.	IY-2 (PSAN-2)	Konverter sinyal pneumatik ke sinyal listrik	Input : 0,2-1 bar Output : 4-20 mA
3.	IV-1	Konverter arus ke tegangan	Input : 4-20 mA Output : 0-5 V
4.	IV-2	Konverter arus ke tegangan	Input : 4-20 mA Output : 0-5 V
5.	VI	Konverter tegangan ke arus	Input : 0-5 V Output : 4-20 mA
6.	DAQ (CPU, DAC, dan ADC)	Pengolah sinyal analog ke digital (ADC) atau sebaliknya dari sinyal digital ke analog (DAC)	Input : 0-5 V Output : 0-5 V
7.	Selektor (relay)	Mengubah mode pengendalian dari digital ke analog atau sebaliknya	

3.2.1 Penambahan Dua Instrumen (*pressure sensor*)

Untuk mengendalikan proses pada Tangki 2, maka diperlukan *pressure sensor*. Sensor yang digunakan berjumlah dua buah, IY-1 (PSAN-1) dan IY-2 (PSAN-2). IY-1 bertindak sebagai konverter sinyal pneumatik ke listrik dari FT-1. Sedangkan IY-2 sebagai konverter sinyal pneumatik ke listrik dari LT-1. Konverter tersebut diperlukan dikarenakan pada Tangki 2, sistem pengendaliannya menggunakan transmisi sinyal pneumatik. Masing-masing sensor tersebut akan mengubah keluaran dari FT-1 dan LT-1 menjadi besaran listrik berupa arus 4-20 mA. Pada *plant*, FT-1 bertindak sebagai *flow sensor*. Sementara LT-1 berfungsi sebagai *level sensor*. Seperti Gambar 3.5, untuk penyadapan, sinyal pneumatik akan diambil dari masukan FR-1/LR-1. Agar sensor dapat bekerja sesuai dengan harapan, maka diperlukan beberapa pengaturan pada parameternya yaitu :

1. Unit : psi
2. Resolusi : 2000
3. Mode operasi output : *auto*
4. Setting output : 1o2o
5. Respon waktu : 2,5 ms

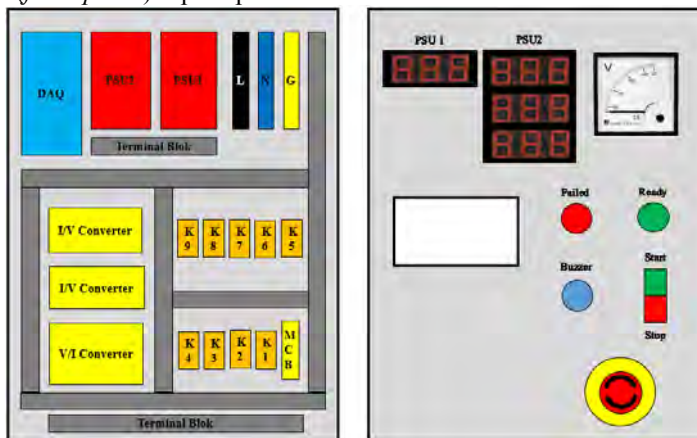
Sensor Autonics ini mempunyai kelinieran yang cukup tinggi. Penyimpangannya hanya sekitar $\pm 2\%$ dari *full scale*. Sedangkan untuk impedansi *output* sebesar $1\text{ k}\Omega$.



Gambar 3.5 Mekanisme Penjadapan Sinyal Pneumatik

3.2.2 Penambahan Panel Kontrol

Panel kontrol ini berukuran 60x80 cm (lebar x tinggi). Panel kontrol tersebut dibuat untuk menampung semua komponen elektrik yang dibutuhkan untuk merealisasikan sistem pengendalian proses. Panel kontrol ini berfungsi sebagai *interface* antara *plant* dan komputer. Panel kontrol ini mempunyai dua bagian, bagian dalam (*plat*) dan bagian luar (*front panel*) seperti pada Gambar 3.6.



Gambar 3.6 Konfigurasi Tata Letak Komponen di Panel Kontrol

Pada bagian dalam terdapat beberapa rangkaian dan komponen elektronik, diantaranya :

1. Dua buah rangkaian I/V *Converter*
2. Rangkaian V/I *Converter*
3. *Data Acquisition* (DAQ) terdiri dari CPU, DAC, dan ADC
4. *Power Supply Unit* (PSU) +5V, +12V, -12V
5. *Power Supply Unit* (PSU) 24 V
6. Tujuh buah Relay 24 VDC 5A
7. Dua buah 220 VAC 7A
8. *Mini Circuit Breaker* (MCB) 2A
9. Terminal Blok tiga buah *Bus Bar* (L,N, dan G).

Sementara untuk bagian luar (*front panel*), terdapat beberapa komponen yaitu :

1. Dua buah lampu indikator 24 volt yaitu *ready* dan *failed*
2. Satu buah *buzzer* 24 volt
3. Satu buah *emergency stop button*
4. Satu buah *start-stop button*
5. Empat buah *digital voltmeter*
6. Satu buah *analog voltmeter*

Seperti pada Gambar 3.4, panel kontrol mempunyai masukan dari PICout, IY-1, dan IY-2. Sedangkan untuk keluaran, panel kontrol hanya mempunyai satu keluaran menuju I/P.

Untuk mekanisme kerja dari panel kontrol dijelaskan sebagai berikut. Tombol *emergency stop* dirangkai seri dengan jala-jala PLN dengan kondisi *normally close*. Sementara tombol *start* digunakan untuk mengaktifkan panel kontrol. Sebaliknya, tombol *stop* digunakan untuk mematikan panel kontrol, dalam hal ini adalah mematikan semua *power supply* (PSU1 dan PSU2). Berbeda halnya dengan tombol *emergency stop*. Tombol tersebut bertugas untuk memutus hubungan listrik antara panel dengan jala-jala PLN. Sementara untuk indikator *failed* dan *ready* belum difungsikan. Sama halnya dengan *buzzer* yang juga belum difungsikan. Nantinya ketiga komponen tersebut berfungsi untuk memberikan status pada operator mengenai kondisi yang terjadi. Kondisi tersebut ada dua, yaitu normal atau gagal. Untuk lebih jelasnya mengenai

diagram pengawatan dapat dilihat pada Lampiran B. Untuk realisasi panel kontrol dapat dilihat pada Gambar 3.7.

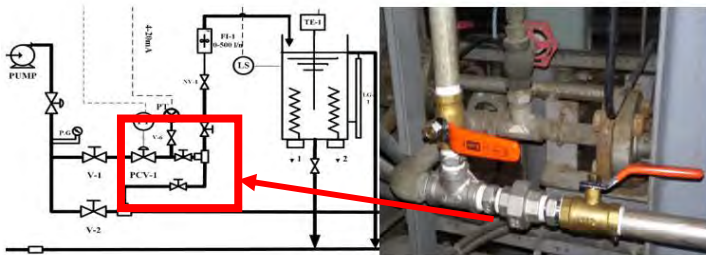


Gambar 3.7 Realisasi Panel Kontrol

3.2.3 Modifikasi Aliran Air dari Tangki 1 ke Tangki 2.

Karena pada Tangki 2 sinyal kontrolnya menggunakan pneumatik penuh, maka *control valve* F(L)CV-1 pada Tangki 2 tidak mempunyai I/P. Padahal, I/P tersebut diperlukan untuk mengendalikan proses Tangki 2 dengan komputer. Oleh karena itu, untuk menyelesaikan permasalahan tersebut, maka jalur aliran air mengalami perubahan. Perubahan tersebut meyangkut perubahan perpipaan pada Tangki 1.

Jalur air Tangki 1 mengalami percabangan ke Tangki 2. Sehingga, Tangki 2 dapat menggunakan *control valve* PCV-1 dari Tangki 1 jika akan dikendalikan menggunakan komputer. Akan tetapi, apabila ingin dikendalikan dengan pneumatik, jalur percabangan pipa ditutup. Agar proses pada T2 dapat dikendalikan dengan F(L)CV-1 lagi. Untuk lebih detil perubahan jalur, dapat dilihat pada Gambar 3.8.



Gambar 3.8 Modifikasi Jalur Flow Air

3.2.4 Penambahan Selektor

Selektor ini bertugas untuk mengubah mode pengendalian dari mode digital ke mode analog atau sebaliknya. Agar dapat mengubah mode tersebut, pada panel terdapat dua buah relay 24 Volt. Relay tersebut akan menyambung/memutus jalur PICout ke PCV-1 atau VI ke PCV-1. Dalam kondisi normal, mode pengendalian yang digunakan adalah mode analog. Apabila mode pengendalian pada mode analog maka PICout akan terhubung dengan PCV-1. Sementara saat mode pengendalian terpilih adalah mode digital, maka PCV-1 akan terhubung dengan VI dari panel seperti pada Gambar 3.9. Tugas tersebut akan dilakukan oleh *relay* K8 dan K9.

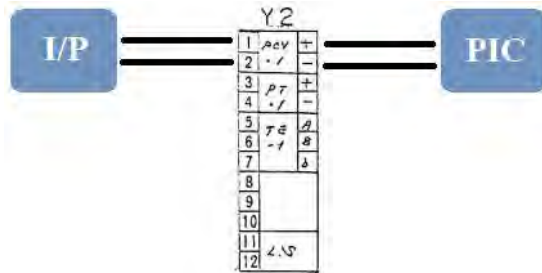
Aktif tidaknya *relay* tergantung perintah dari DAQ. Apabila Digital Input 3 (DI3) aktif maka relay K8 akan aktif. Sementara apabila Digital Input 4 (DI4) aktif maka relay K9 akan aktif.



Gambar 3.9 Skema Jalur Selektor

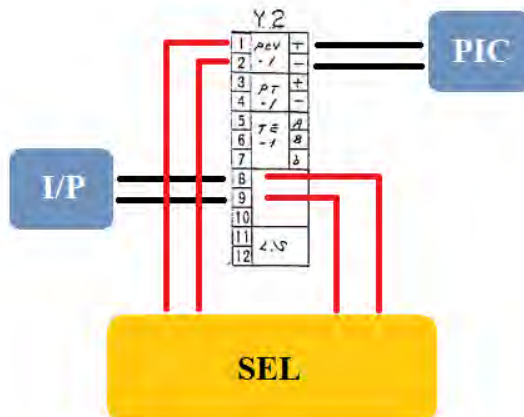
Untuk mekanisme *wiring*-nya adalah seperti berikut. Sambungan I/P dengan PIC terletak pada terminal Y2 nomor 1 dan 2,

seperti terlihat pada Gambar 3.10. Input dari I/P adalah sinyal arus (4-20 mA) DC, sehingga polaritas dari arah arus tidak boleh tertukar. Polaritas positif terletak pada Y2-1 sedangkan untuk negatif pada Y2-2.



Gambar 3.10 Terminal Y2

Kemudian kabel dari I/P pada Y2-1 dan Y2-2 dilepas dan dipindahkan menuju Y2-8 dan Y2-9. Y2-8 untuk polaritas positif dan Y2-9 untuk polaritas negatif seperti pada Gambar 3.11.



Gambar 3.11 Modifikasi *Wiring* Terminal Y2

3.3 Perancangan Perangkat Keras

Perancangan perangkat keras yang akan dibahas meliputi perancangan M-DAQ dan *signal conditioning*. Kedua komponen tersebut merupakan komponen yang dibuat pada Tugas Akhir ini. Kedua komponen tersebut terletak pada panel kontrol.

3.3.1 Data Acquisition (DAQ)

DAQ yang didesain menggunakan konsep modular. Artinya DAQ akan tersusun dari berbagai modul dengan fungsi yang beragam. Masing-masing modul dihubungkan dengan satu bus data seperti pada Gambar 3.12.

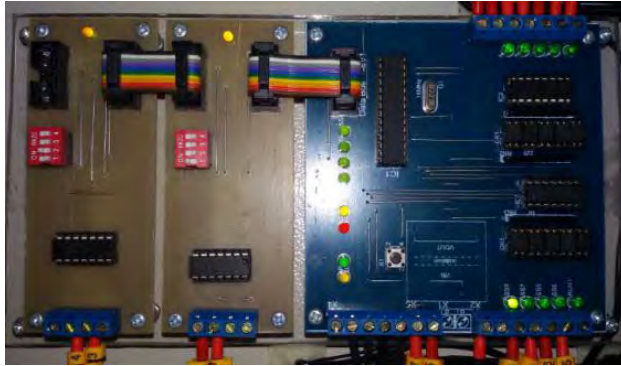
Bus data yang digunakan adalah bus SPI. Pemilihan bus SPI karena kecepatan transfer yang sangat cepat. Komponen utama yang ada pada DAQ ini adalah mikrokontroler ATmega 328P (CPU), *Analog to Digital Converter* (ADC), *Digital to Analog Converter* (DAC).

Sedangkan untuk koneksi dengan komputer, M-DAQ dirancang menggunakan RS-485 *full duplex*. Pemilihan RS-485 karena lebih tahan *noise* dan mampu digunakan untuk transmisi jarak jauh hingga 1.200 meter.

Masing-masing komponen secara berurutan akan dibahas pada Subsubsubbab 3.2.2.1, 3.2.2.2, 3.2.2.3, dan 3.2.2.4. 3.2.2.1. Adapun spesifikasi DAQ yang dihasilkan disajikan pada Tabel 3. 2.

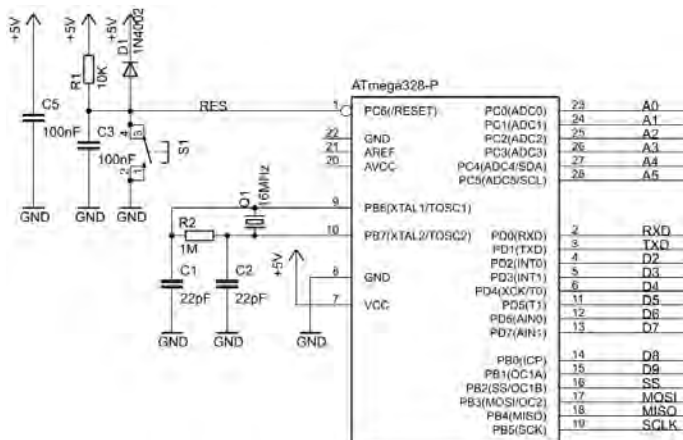
Tabel 3. 2 Spesifikasi DAQ

No	Variabel	Keterangan
1.	Mikokontroler	ATmega328P
2.	<i>Supply</i>	5 V
3.	Tegangan Masukan DAQ (rekomendasi)	0-5V
4.	Tegangan Masukan DAQ (batas)	-0.6-5.6V
5.	Tegangan Output DAQ	-0.3-5.3 V
6.	Jumlah Channel Input Digital	5
7.	Jumlah Channel Output Digital	5
8.	Jumlah Channel Input Analog	4
9.	Jumlah Channel Output Analog	2
10.	Kecepatan Clock	16 MHz
11.	Panjang	160 mm
12.	Lebar	100 mm



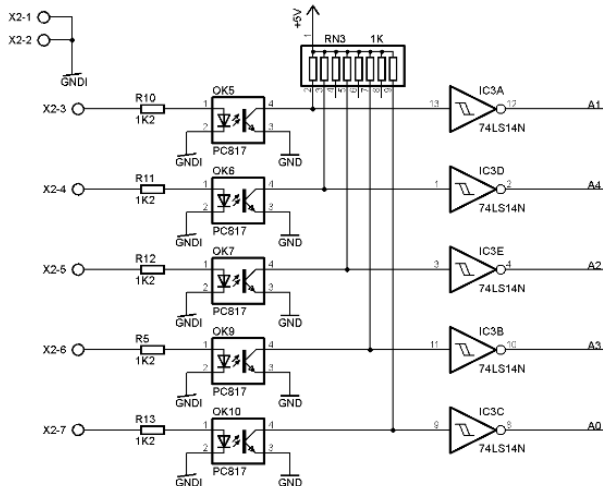
Gambar 3.12 Realisasi DAQ

Melalui CPU komputer dapat mengambil data dari ADC dan mengeluarkan data ke DAC. Pada CPU terdapat 5 *digital input* dan 5 buah *digital output*. Untuk CPU, mikrokontroler yang digunakan adalah ATmega328P dengan minimum sistem seperti Gambar 3. 13. ATmega328P yang digunakan sudah diisi dengan *bootloader* Arduino.



Gambar 3. 13 Minimum Sistem CPU DAQ

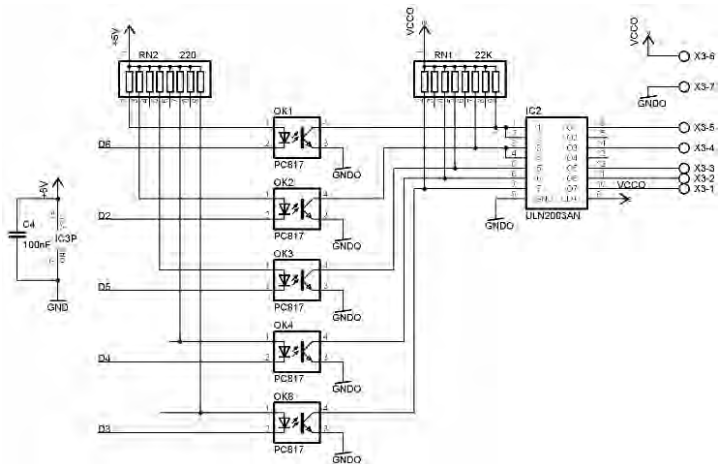
Pada CPU disiapkan tombol reset S1 yang dapat digunakan untuk mereset CPU jika terjadi *error* saat beroperasi. Pada *digital input* digunakan isolator berupa optocoupler untuk memisahkan hubungan elektrik antara DAQ dengan komponen luar. *Digital input* bekerja pada tegangan nominal 24 VDC. Karakteristik digital input adalah *current sink* dengan arus nominal 25 mA. Output dari optocoupler ini didesain dengan konfigurasi aktif *low*. Untuk membalik logika, maka digunakan IC 74LS14, yaitu IC dengan *schmitt trigger*. Output dari IC 74LS14 ini dihubungkan dengan pin *digital input* ATmega328P. *Digital input* dihubungkan menuju pin A0-A5 pada minimum sistem.



Gambar 3. 14 *Digital Input CPU*

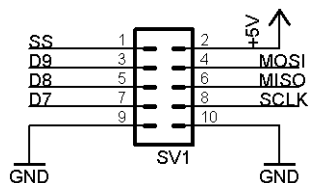
Pada sisi *digital output*, digunakan isolator berupa optocoupler untuk memisahkan hubungan elektrik antara DAQ dengan komponen luar. *Output digital* ini bekerja pada tegangan nominal 24 Volt DC. Karakteristik digital input CPU *current sink* dengan arus maksimal 250mA. Sebagai *interface* antara optocoupler dengan beban, digunakan IC driver ULN2003. ULN 2003 adalah IC driver dengan konfigurasi *open collector*. Artinya tugas ULN2003 hanya sebagai saklar penghubung beban dengan *ground*. Gambar 3. 15 adalah hasil rancangan *interface*

digital output dari DAQ. Pin yang digunakan untuk mengakses *output digital* ini adalah pin berlabel D2-D6. D2-D6 ini terhubung dengan CPU sesuai dengan Gambar 3. 15.



Gambar 3. 15 *Digital Output DAQ*

Sedangkan untuk komunikasi data antara CPU dengan modul-modulnya, digunakan bus SPI. Koneksi bus SPI dari CPU menuju modul menggunakan kabel paralel 10 pin. Berikut adalah konfigurasi pin dari konektor bus SPI dari CPU seperti Gambar 3.16.



Gambar 3.16 Konfigurasi Pin Slave Bus

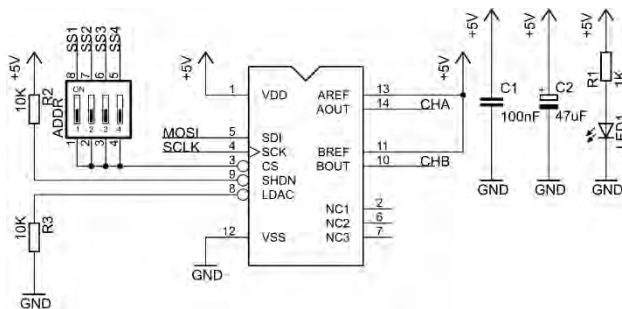
CPU M-DAQ membutuhkan suplai daya 5 Volt DC agar mampu bekerja dengan optimal. Tegangan mininum operasinya adalah 4,75 Volt DC dan tegangan maksimum operasinya adalah 5,25 Volt DC.

3.3.1.2 Digital to Analog Converter (D/A)

Rangkaian D/A *Converter* berfungsi untuk mengubah sinyal kontrol yang telah dihitung di LabVIEW 2013 dengan algoritma kontroler PID Prediktif menjadi tegangan 0-5 V. Rangkaian tersebut mempunyai dua buah *analog channel output* yang dapat dipilih. Pada Tugas Akhir, *channel* yang digunakan adalah *channel 0* yang terhubung dengan IV. Tegangan 0-5 V tersebut akan diubah VI untuk menjadi arus 4-20 mA.

Untuk resolusi dari DAC, rangkaian ini mempunyai resolusi hingga 2^{12} dengan V_{REF} 5V pada masing-masing *output*. Skematik rangkaiannya dapat dilihat pada Gambar 3.18, pin *Serial Data In* (SDI) pada DAC (pin 5) dihubungkan dengan *Master Output Slave Input* (MOSI) dari CPU (pin 17). Sementara, pin CS dari ADC (pin 3) dihubungkan dengan pin *Select Slave* (SS) pada CPU (pin 16). Sedangkan, pin SCLK pada ADC dihubungkan dengan SCLK di CPU.

Untuk memilih slot dari DAC, dapat dipilih melalui *switch* (SS1-SS4). Pada DAQ, slot DAC terletak pada slot satu sehingga SS1 dalam kondisi ON (aktif).



Gambar 3.18 Rangkaian Digital to Analog Converter (DAC)

3.3.1.3 Modul Komunikasi RS485

Modul komunikasi yang digunakan adalah standar RS485 *full duplex* menggunakan dua IC SN75179B. IC pertama satu *board* dengan CPU. Sementara IC kedua dekat dengan komputer. Untuk konfigurasi rangkaian IC pada *board* CPU DAQ dapat dilihat pada Gambar 3.19.



Gambar 3.21 Realisasi Modul Komunikasi RS485

3.3.2 Rangkaian Konverter

Pada panel kontrol, terdapat dua buah konverter, yaitu konverter I/V dan V/I. Rangkaian konverter pertama digunakan untuk mengubah besaran arus 4-20 mA dari sensor ke dalam *range* tegangan 0-5 V DAQ. Sementara rangkain kedua, mengubah *range* tegangan 0-5 V DAQ menjadi *range* arus 4-20 mA sebagai masukan I/P.

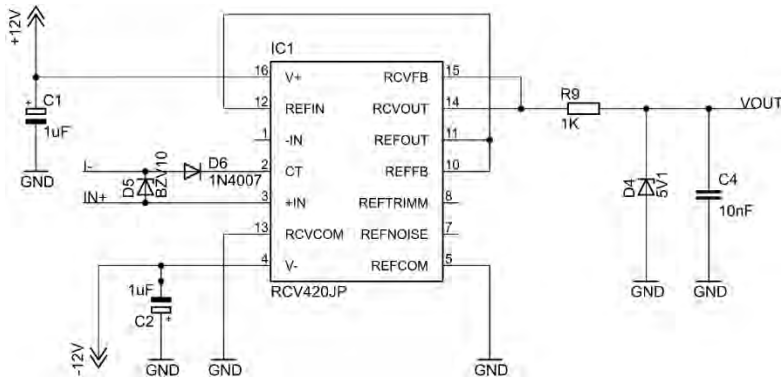
3.3.2.1 I/V Converter

Terdapat dua buah I/V yang kemudian disebut IV-1 dan IV-2. IV-1 berfungsi untuk konverter tegangan dari IY-1. Sementara, IV-2 berfungsi sebagai konverter tegangan dari IY-2. Kedua konverter tersebut menggunakan IC RCV420. IC ini dapat mengubah arus 4-20 mA menjadi tegangan 0-5V secara proporsional. Skematik rangkaian cukup sederhana seperti Gambar 2.13 dan Gambar 3.22, hanya menambahkan kapasitor 1 uF pada kaki pin 4 dan 16. Sementara, pin C_T sebagai IN-. Untuk gain dari IC ini seperti Persamaan (3.1)

$$\frac{V_{out}}{I_{in}} = \frac{5V}{16mA} = 0,3125V / mA \quad (3.1)$$

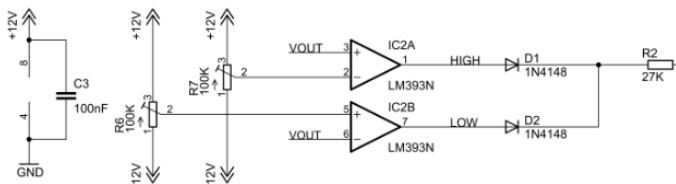
Sementara untuk *offset* sebesar Persamaan (3.2):

$$V_{os} = -(4mA)(0,3125V / mA) = -1,25V \quad (3.2)$$



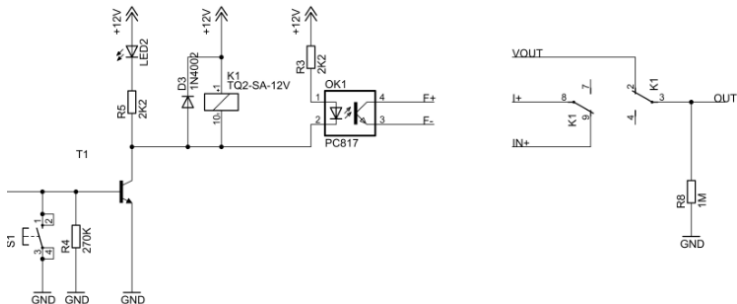
Gambar 3.22 Skematik Rangkaian RCV420

Sebagai tambahan, rangkaian tersebut juga dilengkapi dengan proteksi. Proteksi ini berfungsi untuk melindungi masukan RCV420JP dan ADC. Sesuai dengan rating maksimum RCV420 pada *datasheet*, maka arus masukan tidak boleh lebih dari 40 mA. Sementara, untuk masukan ADC MCP3204 harus dalam *range* -0,6V sampai $V_{DD}+0,6V$. Sehingga diperlukan rangkaian proteksi seperti pada Gambar 3.23 dan Gambar 3.24



Gambar 3.23 Rangkaian Proteksi 1

Pada Gambar 3.23, pin 2 dari LM393 (IC dual komparator) di-*setting* batas maksimum range ADC yaitu 5,6V. Sementara pin 5 di-*setting* batas minimum range ADC yaitu -0,6V. Apabila salah satu/kedua *output* komparator aktif, maka arus akan mengalir pada transistor untuk mengaktifkan relay, seperti pada Gambar 3.24. Sehingga masukan RCV420JP akan terputus.



Gambar 3.24 Rangkaian Proteksi 2



Gambar 3.25 Realisasi IV-1 dan IV-2

Adapun spesifikasi maksimum dari IV adalah seperti Tabel 3.4 :

Tabel 3.4 Spesifikasi IV-1 dan IV-2

No	Variabel	Keterangan
1.	Arus masukan, maksimum	40 mA
2.	Tegangan keluaran	-0,6 V-5,6 V
3.	<i>Span</i>	0.3125
4.	<i>Zero</i>	-1.25
5.	V_{SS}	-12 V
6.	V_{DD}	+12 V

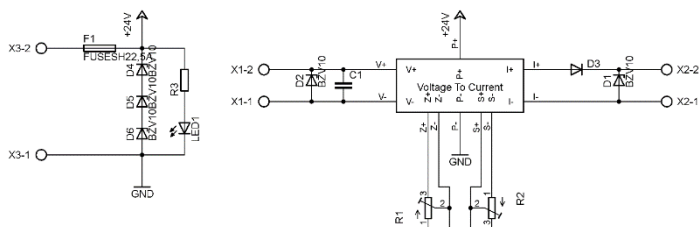
3.3.2.2V/I Converter

Rangkaian konverter yang kedua berupa konverter V/I yang kemudian disebut VI-1. Rangkaian ini berguna untuk mengubah range

output DAQ berupa tegangan 0-5 V ke range input I/P 4-20 mA. IC yang digunakan untuk konverter ini adalah IC Sunyuan ISO-U1-P1-O1. IC ini merupakan jenis *isolated amplifier*. Seperti Gambar 3. 26, Konfigurasi dari rangkaian ini cukup sederhana. Hanya menambahkan dua buah *multiturn* pada kaki pin 8-11 dan 9-10. Pin 8-11 digunakan untuk mengatur *zero* dari VI. Sementara Pin 9-10 digunakan untuk mengatur *span*.

Sebagai proteksi, rangkaian tersebut dilengkapi diode zener D4-D6 untuk membatasi tegangan catu daya 24 Volt. Sementara D2 digunakan untuk melindungi tegangan input. Agar IC ISO-U1-P1-O1 tidak mengalami kegagalan operasi meskipun terjadi kesalahan dalam memberikan input tegangan yang diluar *range* input IC ISO-U1-P1-O1.

Pada bagian output terdapat dua macam proteksi. Proteksi terhadap *reverse polarity*, digunakan dioda penyearah D3. Sedangkan proteksi terhadap arus berlebih ketika terjadi kegagalan pada IC ISO-U1-P1-O1 untuk mengurangi kemungkinan kerusakan pada instrumen I/P digunakan dioda zener D1.



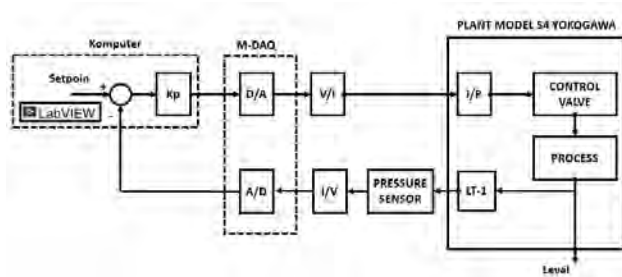
Gambar 3. 26. Rangkaian Skematik VI



Gambar 3.27 Realisasi VI-1

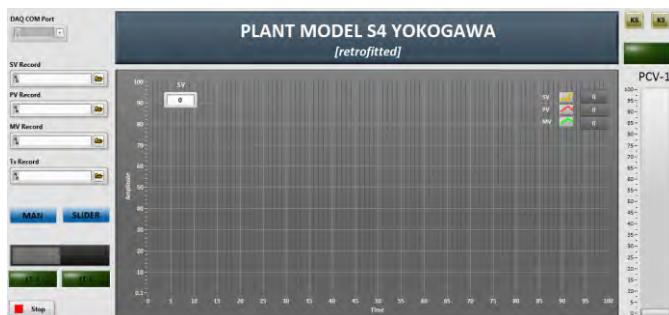
3.4 Identifikasi Sistem

Identifikasi pada Tugas Akhir ini dilakukan secara *closed loop* dengan kontroler proporsional pada komputer. Metode yang digunakan untuk identifikasi adalah metode identifikasi *Strejc*. Melalui analisis identifikasi *Strejc* dengan menggunakan tabel *Strejc*, maka dapat dicari fungsi alih *plant*. Diagram blok untuk identifikasi *plant* ditunjukkan pada Gambar 3.28.



Gambar 3.28 Diagram Blok Identifikasi

Proses identifikasi dilakukan secara *closed loop* dengan memberikan sinyal uji berupa sinyal masukan step 50%. Sinyal uji diberikan saat respon mencapai keadaan *steady state* pada level 10 %. Parameter kontroler proporsional pada komputer yang digunakan sebesar 5 atau setara dengan *proportional band* 20%. Proses pengambilan data dilakukan untuk setiap variasi beban dengan menggunakan NI LabVIEW 2013 dengan waktu *sampling* sebesar 1000 ms seperti pada Gambar 3.29.



Gambar 3.29 Front Panel Identifikasi Plant dengan NI LabVIEW 2013

Gambar 3. 30 menunjukkan *valve* yang digunakan untuk pembebanan level Model S4 Yokogawa.



Gambar 3. 30 *Needle Valve* (NV-2)

Dalam Tugas Akhir ini, *plant* dioperasikan dengan beban yang divariasikan dalam tiga keadaan, yaitu :

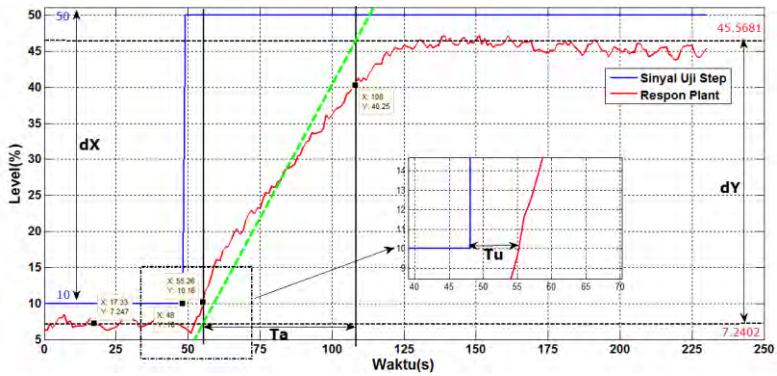
1. Beban minimal : (*Needle Valve*) NV-2 dibuka 3 putaran dari keadaan *full close*
2. Beban nominal : (*Needle Valve*) NV-2 dibuka 5 putaran dari keadaan *full close*.
3. Beban maksimal : (*Needle Valve*) NV-2 dibuka penuh.

3.4.1 Penarikan Garis pada Metode Strejc

Dari masukan *step* yang diberikan untuk tiap kondisi penbebanan, didapatkan respon keluaran level pada beban nominal, maksimal, dan minimal. Respon keluaran *level* pada beban minimal dan pemilihan garis Strejc ditunjukkan pada Gambar 3.31. Garis Strejc pada Gambar 3.31 adalah garis lurus berwarna hijau. Berdasarkan Gambar 3.28, dapat diperoleh nilai parameter *plant*. Nilai *gain overall* dapat dicari sebagai berikut :

$$\Delta Y_{ss} = 38,3279 \% ; \Delta X_{ss} = 40 \%$$

$$K = \frac{\Delta Y_{ss}}{\Delta X_{ss}} = \frac{38,328}{40} = 0,9582$$



Gambar 3.31 Penarikan Garis *Strejc* pada Kurva Respon Beban Nominal

Dan dari Gambar 3.31 juga didapatkan:

$$T_u = 55,26 - 48 = 7,26 \text{ detik}$$

$$T_a = 105 - 55,26 = 49,740 \text{ detik}$$

Maka:

$$\frac{T_u}{T_a} = \frac{7,24}{49,740} = 0,1521$$

Berdasarkan tabel dan prosedur Strejc, didapat :

$$n = 2$$

$$\frac{T_a}{T} = 2,728 \Leftrightarrow T = \frac{T_a}{2,728} = \frac{49,740}{2,728} = 17,5644 \text{ detik}$$

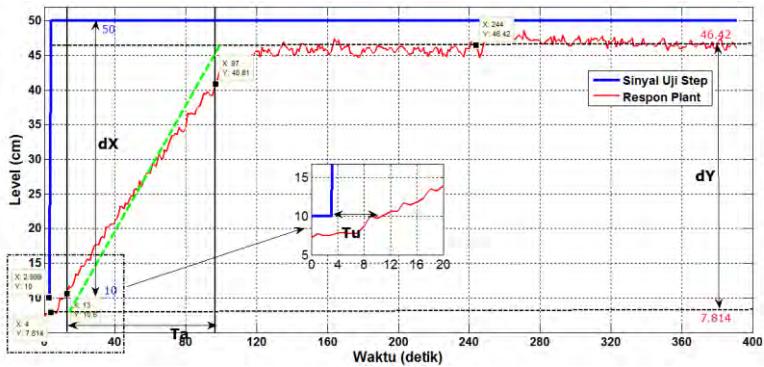
$$T_u' = \left(\frac{T_u}{T_a} \right)_{tabel} T_a \Leftrightarrow T_u' = 0,108(49,740) = 4,9650$$

$$L = T_u - T_u' = 7,24 - 4,9650 = 2,2950$$

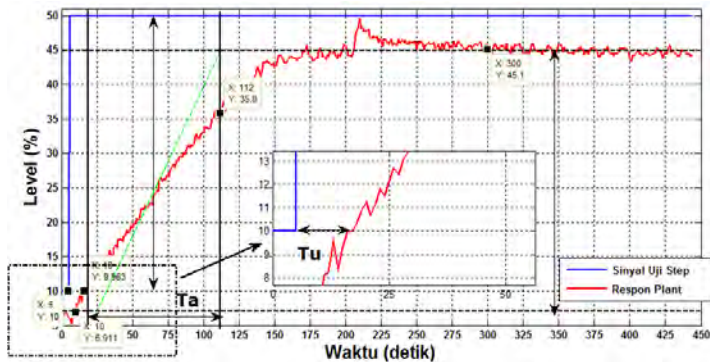
Sehingga model matematika berupa fungsi penghantar *loop* tertutup *plant* untuk kondisi minimal adalah :

$$G(s) = \frac{0.958}{(1 + 17,5644s)^2} e^{-2.2950s}$$

Langkah yang sama dilakukan pada beban nominal dan maksimal. Respon *plant* untuk beban nominal dan beban maksimal dapat dilihat pada Gambar 3.32 dan Gambar 3.33.



Gambar 3.32 Penarikan Garis Strejc pada Kurva Respon Beban Minimal



Gambar 3.33 Penarikan Garis Strejc pada Kurva Respon Beban Maksimal

Langkah identifikasi yang sama dilakukan untuk variasi beban lainnya. Hasil identifikasi *plant* untuk kondisi beban nominal, kondisi beban maksimal, dan kondisi beban minimal diberikan oleh Tabel 3. 5

Tabel 3. 5 Model Plant *Loop* Tertutup

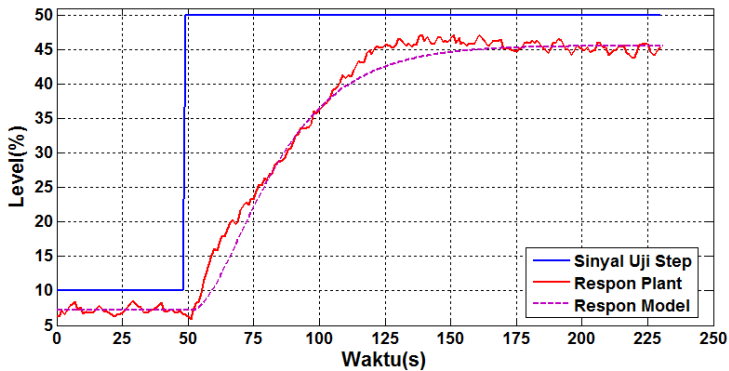
Beban	Model
Minimal	$G(s) = \frac{0,9498e^{-0,2640s}}{(30,905+1)^2}$
	$G(s) = \frac{0,9652e^{-1,2650s}}{(30,905+1)^2}$
	$G(s) = \frac{0,9854e^{-1,3290s}}{(34,2163s+1)^2}$
Nominal	$G(s) = \frac{0,958e^{-1,7750s}}{(19,4040s+1)^2}$
	$G(s) = \frac{0,9582e^{-2,2950s}}{(17,5644s+1)^2}$
	$G(s) = \frac{0,96e^{-2,0870s}}{(18,3002s+1)^2}$
Maksimal	$G(s) = \frac{0,95474e^{-5,126s}}{(33,7351s+1)^2}$
	$G(s) = \frac{0,9838e^{-1,3290s}}{(32,233s+1)^2}$
	$G(s) = \frac{0,963e^{-3,3290s}}{(33,452s+1)^2}$

3.4.2 Validasi Model

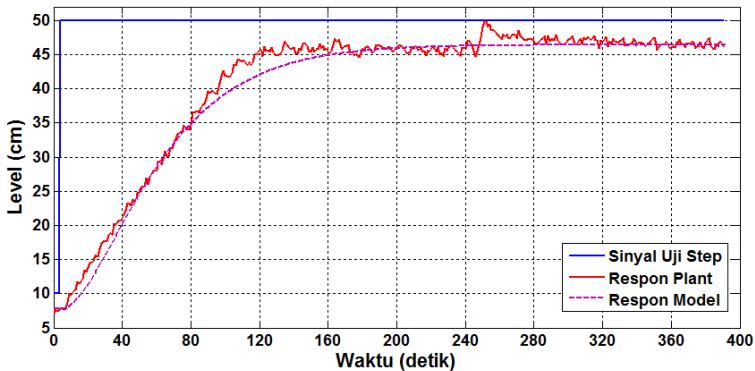
Untuk mengetahui nilai kebenaran dari model plant yang telah dihitung menggunakan metode *Strejc*, maka data percobaan dilakukan validasi untuk setiap variasi beban. Kemudian dicari nilai *Relatif Steady State Error* (RSSE), semakin kecil nilai RSSE maka semakin baik model yang dibuat. Nilai RSSE dengan Persamaan (3.3) merupakan nilai dalam persen dari RMSE pada Persamaan (2.3) dibandingkan dengan *Y steady state* (Y_{ss}) plant.

$$RSSE = \frac{RMSE}{Y_{ss}}.(100\%) \quad (3.3)$$

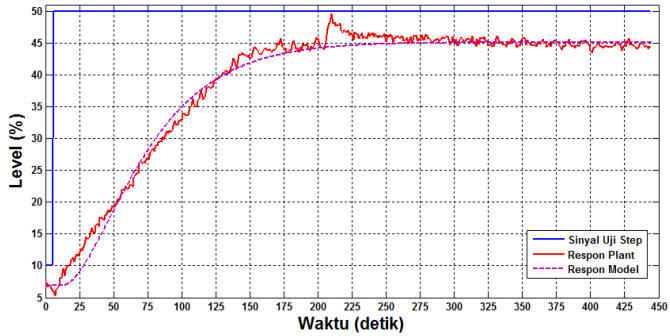
Sesuai dengan Persamaan (3.3) untuk validasi model, didapat nilai RSSE yang disajikan pada Tabel 3.6. Validasi model tersebut penting agar mendapatkan model yang semirip mungkin dengan plant aslinya. Gambar 3.34, Gambar 3.35, dan Gambar 3.36 adalah hasil validasi model dengan nilai RSSE paling kecil jika dibandingkan dengan model lain untuk masing-masing kondisi beban. Ketiga model tersebut yang akan diubah menjadi persamaan *loop* terbuka.



Gambar 3.34 Validasi Model dengan *Plant* untuk Beban Nominal



Gambar 3.35 Validasi Model dengan *Plant* untuk Beban Minimal



Gambar 3.36 Validasi Model dengan *Plant* untuk Beban Maksimal

Tabel 3.6 Nilai RSSE pada Setiap Kondisi Beban

Beban	Model	RSSE (%)
Minimal	$G(s) = \frac{0,9498e^{-0.2640s}}{(30,905+1)^2}$	4,7628
	$G(s) = \frac{0,9652e^{-1.2650s}}{(30,905+1)^2}$	3,2806
	$G(s) = \frac{0,9854e^{-1.3290s}}{(34,2163s+1)^2}$	4,3487
Nominal	$G(s) = \frac{0,958e^{-1.7750s}}{(19,4040s+1)^2}$	6,3907
	$G(s) = \frac{0,9582e^{-2.2950s}}{(17,5644s+1)^2}$	4,8940
	$G(s) = \frac{0,96e^{-2.0870s}}{(18,3002s+1)^2}$	5,3804
Maksimal	$G(s) = \frac{0,95474e^{-5.126s}}{(33,7351s+1)^2}$	3,9210
	$G(s) = \frac{0,9838e^{-1.3290s}}{(32,233s+1)^2}$	5,4430
	$G(s) = \frac{0,963e^{-3.3290s}}{(33,452s+1)^2}$	4,2640

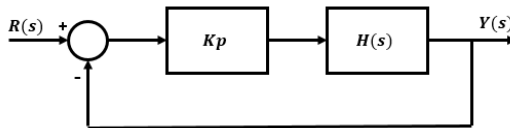
Model *loop* tertutup yang dipilih adalah model yang mempunyai nilai RSSE yang paling kecil. Sehingga dari Tabel 3.6 terpilih model yang disajikan pada Tabel 3.7.

Tabel 3.7 Model *Loop* Tertutup Terpilih

Beban	Model Loop Tertutup
Minimal	$G(s) = \frac{0.9652e^{-1.2650s}}{(30.905s + 1)^2}$
Nominal	$G(s) = \frac{0.9582e^{-2.2950s}}{(17.5644s + 1)^2}$
Maksimal	$G(s) = \frac{0.95474e^{-5.126s}}{(33.7351s + 1)^2}$

3.4.3 Model *Loop* Terbuka

Dari berbagai kondisi beban, model matematika yang didapat, $G(s)$, adalah persamaan loop tertutup dengan penguatan proporsional sama dengan 5. Untuk konfigurasi identifikasi *loop* tertutup dapat dilihat pada Gambar 3.37.



Gambar 3.37 Diagram Blok Identifikasi

$$G = \frac{K_p H(s)}{1 + K_p H(s)} \quad (3.4)$$

Jika dipilih model matematika dari $H(s)$ adalah orde dua dengan *delay* yang diestimasi. Maka $H(s)$ dapat dituliskan kedalam bentuk Persamaan (3.5)

$$H(s) = \frac{Ke^{-\theta s}}{\alpha s^2 + \beta' s + 1} \cong \frac{K(1 - \theta s)}{\alpha s^2 + \beta s + 1} \quad (3.5)$$

Maka sesuai dengan Persamaan (3.5), dapat ditulis ke bentuk Persamaan (3.6):

$$\begin{aligned}
 G(s) &= \frac{\frac{K_p K(1-\theta s)}{\alpha s^2 + \beta s + 1}}{1 + \frac{K_p K(1-\theta s)}{\alpha s^2 + \beta s + 1}} \\
 G(s) &= \frac{K_p K(1-\theta s)}{\alpha s^2 + (\beta - K_p K \theta) s + (1 + K_p K)} \\
 G(s) &= \frac{\frac{K_p K}{(1 + K_p K)}(1-\theta s)}{\frac{\alpha}{(1 + K_p K)} s^2 + \frac{(\beta - K_p K \theta)}{(1 + K_p K)} s + 1} \quad (3.6)
 \end{aligned}$$

Atau

$$G(s) = \frac{K'(1-\theta s)}{\alpha' s^2 + \beta' s + 1} \quad (3.7)$$

Struktur Persamaan (3.6) ekivalen dengan persamaan model matematika yang ditunjukkan oleh Persamaan (3.5) yang merupakan persamaan loop terbuka. Sehingga :

$$\begin{aligned}
 K' &= \frac{K_p K}{(1 + K_p K)} \Leftrightarrow K = \frac{K'}{K_p(1 - K')} \\
 \alpha' &= \frac{\alpha}{(1 + K_p K)} \Leftrightarrow \alpha = \alpha'(1 + K_p K) \\
 \beta' &= \frac{\beta - K_p K \theta}{(1 + K_p K)} \Leftrightarrow \beta = \beta'(1 + K_p K) + K_p K \theta
 \end{aligned}$$

Dengan $H(s)$ adalah model loop terbuka. Sedangkan $G(s)$ adalah model *loop* tertutup. K_p adalah penguatan proporsional. Hasil perhitungan model *plant loop* terbuka dapat dilihat pada Tabel 3.8. Model *loop* terbuka ini adalah model matematika *plant* yang akan dikontrol.

Tabel 3.8 Model *Loop* Terbuka

Beban	Model <i>Loop</i> Terbuka
Minimal	$H(s) = \frac{-7,017s + 5,547}{(27446s^2 + 1811s + 1)}$
Nominal	$H(s) = \frac{-10,52s + 4,585}{(7381s^2 + 893s + 1)}$
Maksimal	$H(s) = \frac{-21,61s + 4,215}{(25123s^2 + 883,5s + 1)}$

Karena kontroler PID Prediktif menggunakan model diskrit, maka model *plant* pada Tabel 3.8 diubah kedalam bentuk diskrit seperti pada Tabel 3.9 menggunakan Persamaan (2.2).

Tabel 3.9 Model Diskrit Plant

Beban	Model <i>Loop</i> Terbuka
Minimal	$H(z) = \frac{-0,0001485z + 0,0003441}{(z^2 - 1,9936s + 0,9361)}$
Nominal	$H(z) = \frac{-0,001044z + 0,001629}{(z^2 - 1,886s + 0,886)}$
Maksimal	$H(z) = \frac{-0,0007622z + 0,000927}{(z^2 - 1,965z + 0,9645)}$

3.5 Perancangan Kontroler

Model matematika dari kontroler PID secara umum adalah seperti Persamaan (3.8) :

$$u(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt} \quad (3.8)$$

Dalam bentuk diskrit, Persamaan (3.8) menjadi Persamaan (3.9):

$$u(k) = K_p e(k) + K_i \sum_{n=0}^k e(n) + K_d (e(k) - e(k-1)) \quad (3.9)$$

Sehingga untuk perubahan sinyal kontrol menjadi Persamaan (3.10)

$$\Delta u(k) = K_p (e(k) - e(k-1)) + K_i e(k) + K_d (e(k) - 2e(k-1) - e(k-2))$$

$$\Delta u(k) = [K_p + K_i + K_d]e(k) + [-K_i - 2K_d]e(k-1) + K_d e(k-2) \quad (3.10)$$

Untuk menyederhanakan Persamaan (3.10), maka diperoleh Persamaan (3.11)

$$\Delta u(k) = q_0 e(k) + q_1 e(k-1) + q_2 e(k-2) \quad (3.11)$$

Dengan nilai,

$$\left. \begin{aligned} q_0 &= [K_p + K_i + K_d] \\ q_1 &= [-K_i - 2K_d] \\ q_2 &= [K_d] \end{aligned} \right\}$$

Penjelasan diatas sebagai dasar membuat PID Prediktif. Karena kontroler PID Prediktif merupakan kontroler *Model Predictive Control* (MPC) yang dibuat menyerupai kontroler PID. Melalui persamaan perubahan sinyal kontrol pada Persamaan (2.28) , didapat Persamaan (3.12)

$$\Delta u(k) = \left((\Phi^T \Phi + R)^{-1} \Phi^T \right) \left(R_s - [F_0 \quad F_1 \quad F_2] \begin{bmatrix} y(k) \\ y(k-1) \\ y(k-2) \end{bmatrix} - G \Delta u(k-1) \right) \quad (3.12)$$

Misalkan

$$K = \left((\Phi^T \Phi + R)^{-1} \Phi^T \right)$$

Sehingga Persamaan (3.12) menjadi Persamaan (3.13)

$$\Delta u(k) = K \begin{bmatrix} 1 & \dots & Np \end{bmatrix}^T r(k) - K \begin{bmatrix} f_{1,0}y(k) + f_{1,1}y(k-1) + f_{1,2}y(k-2) \\ f_{2,0}y(k) + f_{2,1}y(k-1) + f_{2,2}y(k-2) \\ \vdots \\ f_{Np,0}y(k) + f_{Np,1}y(k-1) + f_{Np,2}y(k-2) \end{bmatrix} - KG \Delta u(k-1)$$

$$\begin{aligned} \Delta u(k) = & \sum_{i=1}^{Np} k_i r(k) \\ & - \left[\sum_{i=1}^{Np} k_i f_{i,0} y(k) + \sum_{i=1}^{Np} k_i f_{i,1} y(k-1) + \sum_{i=1}^{Np} k_i f_{i,2} y(k-2) \right] - k_i g_i \Delta u(k-1) \end{aligned} \quad (3.13)$$

Dengan masing-masing nilai *gain* seperti pada Persamaan (3.14), (3.15), (3.16), dan (3.17)

$$q_{ff} = \sum_{i=1}^{Np} k_i \quad (3.14)$$

$$q_0 = \sum_{i=1}^{Np} k_i f_{i,0} \quad (3.15)$$

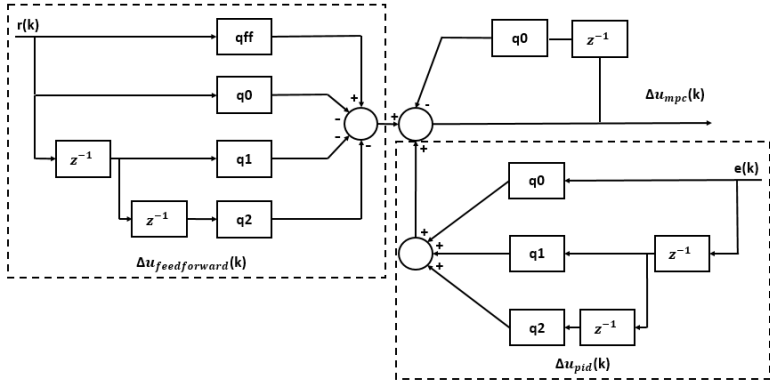
$$q_1 = \sum_{i=1}^{Np} k_i f_{i,1} \quad (3.16)$$

$$q_2 = \sum_{i=1}^{Np} k_i f_{i,2} \quad (3.17)$$

$$q_u = \sum_{i=1}^{Np} k_i g_i \quad (3.18)$$

Jika diuraikan menjadi PID Prediktif, didapat Persamaan (3.19)

$$\begin{aligned} \Delta u(k) = & q_{ff} r(k) - \left[q_0 y(k) + q_1 y(k-1) + q_2 y(k-2) \right] - q_u \Delta u(k-1) \\ \Delta u(k) = & \overbrace{q_{ff} r(k) - q_0 r(k) - q_1 r(k-1) - q_2 r(k-2)}^{\Delta u_{feedforward}} \\ & + \overbrace{q_0 e(k) + q_1 e(k-1) + q_2 e(k-2)}^{\Delta u_{PID}} - q_u \Delta u(k-1) \end{aligned} \quad (3.19)$$



Gambar 3.38 Struktur Kontroler PID Prediktif

Berdasarkan penjelasan diatas, kontroler PID Prediktif terdiri dari *gain* yang didapat dari *Model Predictive Control* (MPC). Kontroler yang akan dirancang mengacu pada Subbab 2.3 dan 2.4. Dalam perancangannya, kontroler PID Prediktif disusun melalui beberapa prosedur desain :

1. Ubah fungsi alih *plant* kontinu ke diskrit seperti pada dengan (2.2) *time sampling* sama dengan identifikasi *plant* yaitu 1 sekon.
2. Hitung matriks \mathbf{F} , \mathbf{G} , Φ seperti pada dengan iterasi ssebanyak N_p . Dalam perancangan kontroler, N_p dipilih sama dengan 200.
3. Selesaikan (3.13) untuk mendapat perubahan sinyal kontrol (Δu) dengan prinsip *receding horizon*.
4. Hitung *gain-gain* dari PID Prediktif melalui Persamaan (3.14), (3.15), (3.16), (3.17), dan (3.18).
5. Pilih variabel tuning rw yang sesuai agar gain dari PID Prediktif mempunyai respon yang diinginkan.

Berdasarkan hasil dari identifikasi, model yang didapat adalah dalam bentuk kontinu pada Persamaan (3.20) :

$$H(s) = \frac{-10,52s + 4,585}{(7381s^2 + 893s + 1)} \quad (3.20)$$

Model kontinu tersebut diubah menjadi bentuk diskrit sehingga menjadi Persamaan (3.21) sebagai berikut :

$$\frac{Y(z)}{U(z)} = \frac{b_0 z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2} = \frac{-0,001044z + 0,001629}{(z^2 - 1,886z + 0,886)} \quad (3.21)$$

Langkah selanjutnya adalah menghitung matriks **F**, **G**, **Φ**. Sebelum mencari ketiga matriks tersebut, dicari dahulu polinomial dari Persamaan (3.21). Sehingga menghasilkan Persamaan (3.22).

$$\begin{aligned} A(z) &= (1 - z^{-1})(1 - 1,886z^{-1} + 0,886z^{-2}) \\ A(z) &= 1 - 2,886z^{-1} + 2,7720z^{-2} - 0,886z^{-3} \\ b(z) &= -0,001044z^{-1} + 0,001629z^{-2} \end{aligned} \quad (3.22)$$

Pada perancangan kontroler, horizon prediksi diambil sebanyak 200 langkah. Namun sebagai contoh perhitungan, horizon dipilih 3 langkah ke depan.

$$\begin{aligned} y_{k+1} - 2,886y_k + 2,772y_{k-1} - 0,886y_{k-2} &= -0,001044\Delta u_k + 0,001629\Delta u_{k-1} \\ y_{k+2} - 2,886y_{k+1} + 2,772y_k - 0,886y_{k-1} &= -0,001044\Delta u_{k+1} + 0,001629\Delta u_k \\ y_{k+3} - 2,886y_{k+2} + 2,772y_{k+1} - 0,886y_k &= -0,001044\Delta u_{k+2} + 0,001629\Delta u_{k+1} \end{aligned}$$

Sehingga dihasilkan matriks **F**, **G**, **Φ** seperti berikut :

$$\begin{aligned} \Phi &= \begin{bmatrix} -0,0010 & 0 & 0 \\ -0,0014 & -0,0010 & 0 \\ -0,0011 & -0,0014 & -0,0010 \end{bmatrix}; \\ \mathbf{F} &= \begin{bmatrix} 2,8860 & -2,7720 & 0,8860 \\ 5,5570 & -7,1140 & 2,5570 \\ 8,9234 & -12,8470 & 4,9235 \end{bmatrix} \\ \mathbf{G} &= \begin{bmatrix} 0,0016 \\ 0,0047 \\ 0,0091 \end{bmatrix} \end{aligned}$$

Ketiga matriks tersebut disubstitusikan ke Persamaan (3.12). Dengan memilih $rw=0,1$ didapat *gain* dari PID Prediktif sebagai berikut :

$$\begin{aligned} q_{ff} &= -0,003 \\ q_0 &= -0,0174 \\ q_1 &= 0,0227 \\ q_2 &= -0,0084 \\ q_u &= -0,0000154 \end{aligned}$$

Untuk lebih jelas mengenai prosedur desainnya, algoritma dari kontroler dapat dilihat pada Lampiran D. Untuk hasil dari perhitungan didapat gain PID Prediktif seperti Tabel 3.10.

Tabel 3.10 Variasi Gain PID Prediktif dari *Tuning* (*rw*)

No.	Variabel Tuning	Gain PID Prediktif				
		q_r	q_o	q_1	q_2	q_u
1.	<i>rw</i> =50	0,139	67,917	-126,89	59,11	0,109
2.	<i>rw</i> =100	0,098	56,799	-106,27	49,57	0,091
3.	<i>rw</i> =500	0,044	37,586	-70,50	32,96	0,061

Tabel 3.11 Variasi Gain PID Berdasarkan *Tuning* (*rw*)

No.	Variabel Tuning	PID		
		K_p	K_i	K_d
1.	<i>rw</i> =50	8,6699	0,1388	59,11
2.	<i>rw</i> =100	7,1351	0,0984	49,57
3.	<i>rw</i> =500	4,5830	0,0442	32,96

3.6 Perancangan Perangkat Lunak

3.6.1 Protokol CPU M-DAQ

Agar dapat berkomunikasi dengan komputer, maka dibutuhkan sebuah *software* yang mengatur perjanjian komunikasi data antara komputer dan modul CPU. Perjanjian komunikasi data inilah yang kemudian disebut dengan protokol. Protokol ini mengatur kode-kode yang dikirim dan diterima saat komputer ingin mengambil data melalui CPU.

Komunikasi data yang digunakan adalah. Komunikasi UART bersifat *asynchronous*, sehingga kecepatan transfer data harus sesuai antara M-DAQ dan komputer. M-DAQ menggunakan kecepatan transfer data maksimal yang mampu ditangani oleh minimum sistem Atmega328P, yaitu 115200 *baud per second*.

Protokol tersebut didesain dengan format paket data 6 *byte* disetiap mengirim dan menerima data. Paket data yang dikirim menuju DAQ dinamakan *COMMAND*. Sedangkan paket data yang diterima dari DAQ dinamakan *RESPONSE*. Susunan frame dari paket data ini dijelaskan pada Tabel 3.12 dan Tabel 3.13, yang menggambarkan *frame command* dan *frame response*.

Tabel 3.12 Paket *Command*

B0	B1	B2	B3	B4	B5
FC	SN	C0	C1	C2	CRC

FC : *Function Code*

SN : *Slot Number*

C0 : *I/O Configuration Byte [0]*

C1 : *I/O Configuration Byte [1]*

C2 : *I/O Configuration Byte [2]*

CRC : *Cylic Redundancy Check*

Tabel 3.13 Paket *Response*

B0	B1	B2	B3	B4	B5
S0	S1	S2	R1	R2	CRC

S0 : *Status Byte [0]*

S1 : *Status Byte [1]*

S2 : *Status Byte [2]*

R1 : *Response Byte [1]*

R2 : *Response Byte [2]*

CRC : *Cylic Redundancy Check*

Function code adalah kode yang digunakan untuk memberikan perintah pada CPU. Slot number merupakan slot yang ingin diakses. Sedangkan C0-C2 adalah *byte* yang dikirimkan sebagai parameter pengambilan data. CRC adalah *byte* yang digunakan untuk menguji kevalidan command dari komputer. Jika CRC benar, maka Status byte S0-S1 berisi “ACK” sebaliknya jika salah, maka berisi “NAK”. R0 dan R1 merupakan data yang dikirimkan kembali menuju komputer oleh CPU yang kemudian diolah pada *software* LabVIEW pada Tabel 3.14 dan

Tabel 3. 15**Tabel 3.14** Commad ke M-DAQ

Function	FC	SN	C0	C1	C2
	0	1	2	3	4
Read ADC-SE	0x00	Slot	Channel	x	x
Read ADC-PD	0x01	Slot	Channel	x	x
Write DAC-G1	0x02	Slot	Channel	W1	W0
Write DAC-G2	0x03	Slot	Channel	W1	W0
Digital Output-SC	0x04	0x00	Channel	State	x
Digital Output-MC	0x05	0x00	GC	GS	x
Digital Input-SC	0x06	0x00	Channel	x	x
Digital Input-MC	0x07	0x00	GC	x	x

Tabel 3. 15 *Response* dari M-DAQ

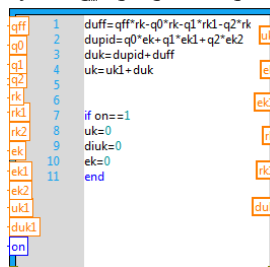
Function	S0-S2	C1	C2
	0-2	3	4
Read ADC-SE	ACK/NAK	AH	AL
Read ADC-PD	ACK/NAK	AH	AL
Write DAC-G1	ACK/NAK	D0	D1
Write DAC-G2	ACK/NAK	D0	D1
Digital Output-SC	ACK/NAK	CH	ST
Digital Output-MC	ACK/NAK	CH	ST
Digital Input-SC	ACK/NAK	CH	ST
Digital Input-MC	ACK/NAK	CH	ST

Implementasi protokol tersebut pada software Arduino IDE 1.0.6 terlampir pada Lampiran C.

3.6.2 Simulasi

Sebelum diimplementasikan pada *plant* nyata Plant Model Yokogawa Tipe S4, algoritma kontrol hasil desain disimulasikan terlebih dahulu dengan menggunakan NI LabVIEW 2013. Diagram blok sistem kontrol hasil desain untuk simulasi ditunjukkan pada Gambar 3.39.

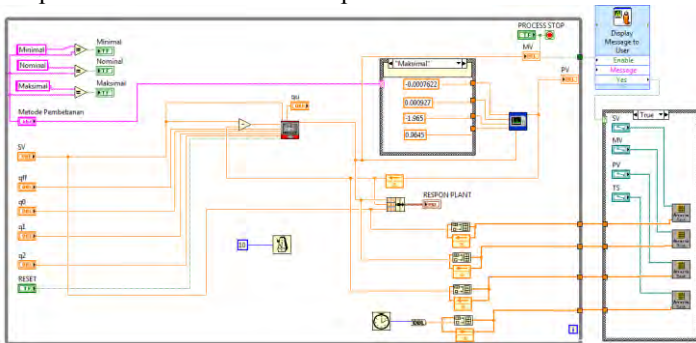
Pada simulasi ini dirancang *subVI* dari kontroler PID Prediktif yaitu *PID Prediktif.vi*. *SubVI* tersebut berisi algoritma dari kontroler dalam bentuk *Mathscript*. Dengan menggunakan blok fungsi *Mathscript*, dapat meringkas diagram blok yang ada di simulasi. *Mathscript* ini mirip dengan M-file dari MATLAB. Kontroler PID Prediktif dalam *Mathscript* dipengaruhi oleh *gain* yang didapat dari perhitungan pada perancangan kontroler Sub Bab 3.3.4 yaitu q_0 , q_1 , dan q_2 .



Gambar 3.39 Algoritma PID Prediktif di *Mathscript*

Untuk mendapat nilai dari *gain-gain* tersebut, digunakan bantuan M-File dari Matlab 2013. Untuk potongan dari programnya dapat dilihat

pada Lampiran D. Secara lengkap diagram blok untuk simulasi dapat dilihat pada Gambar 3.40 atau Lampiran E.

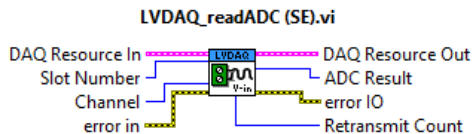


Gambar 3.40 Diagram Blok Simulasi

3.6.2 Implementasi

3.6.2.1 SubVI Read ADC

Berbeda dengan simulasi, pada implementasi terdapat dua buah *subVI* tambahan yaitu, *readADC.vi* dan *writeDAC.vi*. *SubVI* pertama digunakan untuk mengolah data dari *Analog to Digital Converter* (ADC) seperti pada Gambar 3.41.



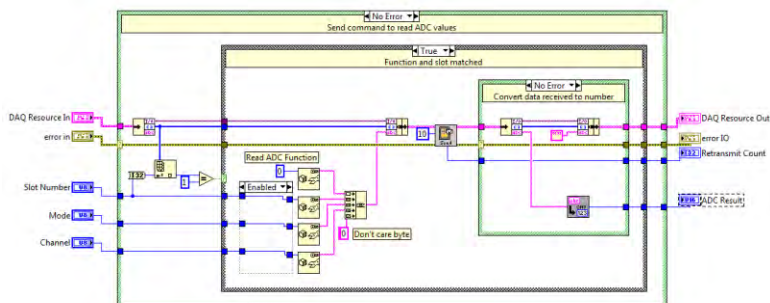
Gambar 3.41 SubVI Membaca Nilai ADC

SubVI pada Gambar 3.41. adalah *SubVI* yang akan digunakan untuk mengambil data *analog input* dari modul analog input M-DAQ. *SubVI* ini sudah menangani permasalahan protokol komunikasi data dengan M-DAQ. Sehingga cukup dengan memberikan informasi slot modul analog input serta *channel* yang akan diakses. *SubVI* ini akan memberikan hasil baca analog input dengan format *unsigned integer 16 bit* (U16).

Pada *subVI* tersebut terdapat beberapa pilihan yaitu letak slot ADC (slot 1-4), nomor *channel* (*channel* 0-3), dan mode ADC (*single ended* atau *pseudo differential*). ADC pada DAQ yang dibuat mempunyai empat buah input analog, sehingga *channel*-nya ada empat buah seperti pada Gambar 3.42 dan Gambar 3.43.



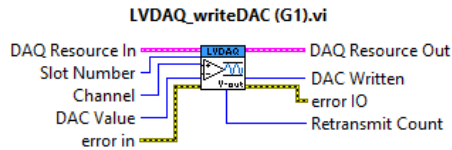
Gambar 3.42 Tampilan Front Panel *readADC.vi*



Gambar 3.43 Blok Diagram *readADC.vi*

3.6.2.2 SubVI Write ADC

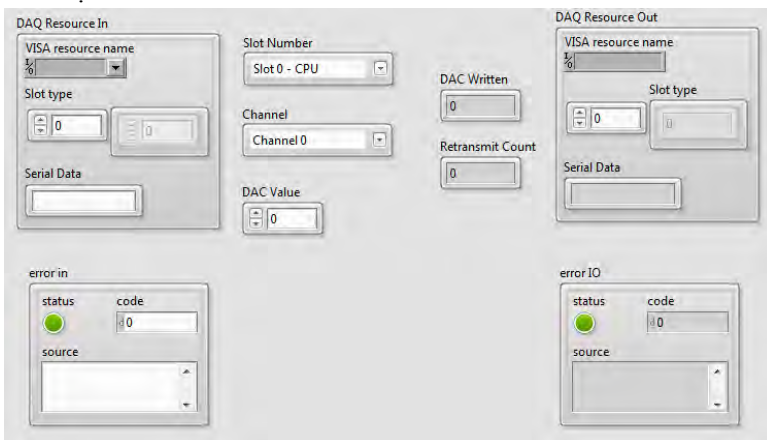
Untuk *subVI* kedua, digunakan untuk mengirimkan sinyal kontrol yang telah didapat dari kontroler PID Prediktif. Sinyal kontrol tersebut dikirim menuju *Digital to Analog Converter* (DAC) untuk diubah menjadi tegangan 0-5 volt.



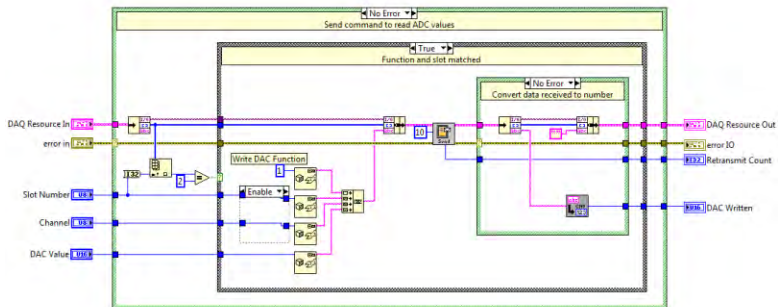
Gambar 3.44 *SubVI* Menulis Nilai pada DAC

SubVI pada Gambar 3.44 adalah SubVI yang akan digunakan untuk mengirimkan data nilai analog output menuju M-DAQ. SubVI ini sudah menangani permasalahan protokol komunikasi data dengan M-DAQ. Sehingga cukup dengan memberikan informasi slot modul analog output serta *channel* yang akan diakses. Data yang dikirimkan menuju M-DAQ harus dalam format *unsigned integer 16 bit* (U16).

Sama seperti *subVI readADC.vi*, *writeDAC.vi* juga mempunyai beberapa pilihan yaitu nomor slot (slot 1-4). Untuk *channel*, hanya ada dua pilihan yaitu *channel 0* dan *2* karena DAC tersebut hanya mempunyai dua *ouput* analog. Pada desain DAQ, slot 0 selalu ditempati CPU atau mikrokontroler ATmega32/ATmega328P



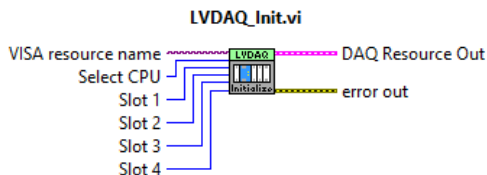
Gambar 3.45 Tampilan Front Panel *wireDAC.vi*



Gambar 3.46 Diagram Blok *writeDAC.vi*

3.6.2.3 Inisialisasi Komunikasi

Sementara itu, untuk inisialisasi program, juga terdapat *subVI* untuk komunikasi data secara *serial* antara komputer dengan *hardware* yaitu *Data Acquisition* (DAQ).



Gambar 3.47 SubVI Inisiasi Program

SubVI pada Gambar 3.47 adalah yang pertama harus dipanggil saat ingin memulai komunikasi data dengan M-DAQ. Pada SubVI tersebut berisi konfigurasi *hardware* dan letak COM dari M-DAQ. Konfigurasi *hardware* meliputi pemilihan jenis CPU dan jenis modul yang dimasukkan pada masing-masing slot

Protokol komunikasi data yang dibuat kali ini mirip dengan komunikasi data pada *Labview Interface for Arduino* (LIFA). Hanya saja, komunikasi ini secara khusus dibuat untuk DAQ yang penulis buat. Protokol komunikasi tersebut menggunakan fasilitas komunikasi data serial yang telah ada di LabVIEW yaitu *visa* seperti pada Gambar 3.48 dan Gambar 3.49.

-Halaman ini sengaja dikosongkan-

BAB IV

PENGUJIAN DAN ANALISA

Pada Bab III telah membahas perancangan sistem untuk pengaturan level air pada Plant Model S4 Yokogawa tipe S4. Pada bab ini akan diuraikan mengenai pengujian sistem, simulasi, dan implementasi kontroler PID Prediktif yang telah dibahas pada Bab III. Kemudian hasil simulasi dan implementasi akan dianalisis.

4.1 Pengujian Komponen Sistem

4.1.1 Pengujian *Pressure Sensor* (IY-1 dan IY-2)

Sensor ini berfungsi untuk mengubah sinyal pneumatik menjadi sinyal listrik. Sensor ini bisa dikatakan sebagai sensor *level/flow* digital. Pengujian ini dilakukan dengan cara memberi tegangan *supply* 24 V dan memvariasi masukan *level/flow* dari FT-1/LT-1.

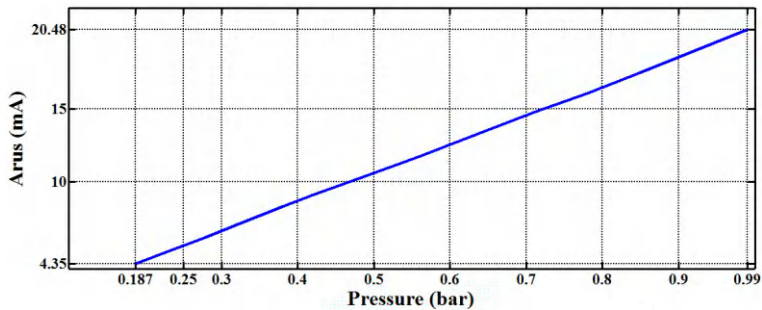
4.1.1.1 *Pressure Sensor-1* (IY-1)

Kalibrasi IY-1 dilakukan dengan menentukan batas atas dan bawah *range* kerja proses *flow*. Proses *flow* berkerja dengan *range* 0-500 l/h. Sehingga batas bawah adalah 0 l/h sedangkan untuk batas atas adalah 500 l/h.

Penentuan batas bawah ditentukan ketika tidak ada aliran air yang mengalir, *flow* = 0. Kemudian mengamati tekanan yang output FT-1 pada indikator IY-1. Diperoleh data bahwa tekanan output FT-1 berada disekitar $\pm 0,187$ bar.

Sementara itu, untuk penentuan batas atas dilakukan dengan membuka PCV-1. Kemudian mengamati *flow* melalui FR-1. PCV-1 dibuka hingga *flow* pada indikator FR-1 mencapai nilai maksimum, yaitu *flow* = 500 l/h. Kemudian mengamati tekanan output dari FT-1 pada indikator IY-1. Diperoleh data bahwa tekanan output FT-1 berada disekitar $\pm 0,930$ bar.

Dari grafik pada Gambar 4.1 terlihat bahwa hubungan antara arus dan *pressure* sangat linier. Dengan nilai minimum (*pressure*=0,187 bar, arus=4,35 mA) untuk *flow* 0 l/h. Sedangkan nilai maksimum (*pressure*=0,930 bar, arus=20,48 mA) untuk *flow* 500 l/h. Sehingga hubungan antara arus dan *pressure* adalah proporsional.

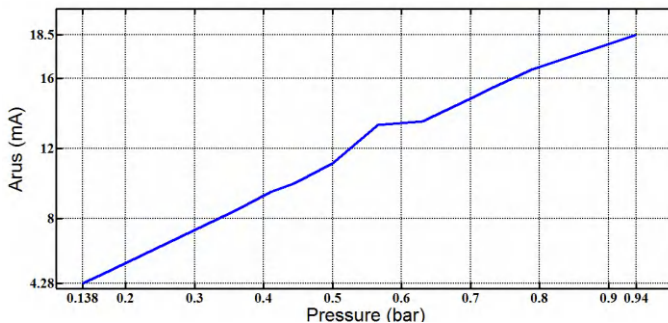


Gambar 4.1 Fungsi Arus terhadap Pressure IY-1

4.1.1.2 IY-2

Sama seperti pengujian pada IY-1. Pada IY-2, penentuan batas bawah dilakukan dengan mengosongkan tanki T2 hingga bacaan pada LR-1 menunjukkan level = 0 mm. Kemudian mengamati tekanan output LT-1 pada indikator IY-2. Diperoleh data bahwa tekanan output LT-1 berada disekitar $\pm 0,138$ bar.

Penentuan batas atas dilakukan dengan cara mengisi tanki hingga penuh. Tanda bahwa tanki terisi penuh dapat dilihat pada indikator LR-1 yang menunjukkan level berada pada nilai 500 mm. Kemudian mengamati tekanan output dari LT-1 pada indikator IY-2. Diperoleh data bahwa tekanan output FT-1 berada disekitar $\pm 0,940$ bar.



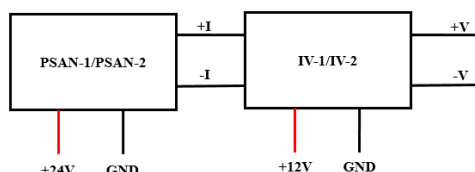
Gambar 4.2 Fungsi Arus terhadap Pressure IY-2

Sama halnya dengan IY-1. Dari grafik pada Gambar 4.1 dan Gambar 4.2 terlihat bahwa hubungan antara arus dan *pressure* sangat linier. Dengan nilai minimum (*pressure*=0,138 bar, arus=4,28 mA) untuk *level* 0 cm. Sedangkan nilai maksimum (*pressure*=0,94 bar, arus=18.5 mA) untuk *level* 50 cm. Sehingga hubungan antara arus dan *pressure* adalah proporsional. Kedua besaran tersebut yaitu *level* dan *flow* dikalibarsi dengan nilai yang ditampilkan oleh FR-1 dan LR-1.

4.1.2 Pengujian Rangkaian Konverter

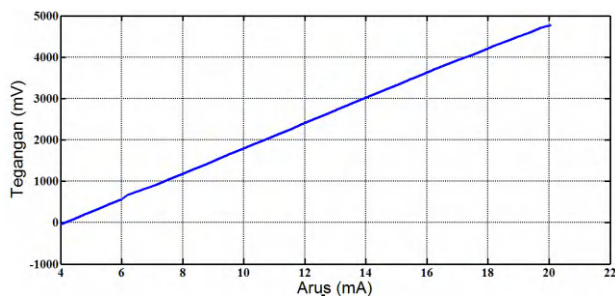
4.1.2.1 Pengujian I/V Converter (IV-1 dan IV-2)

Terdapat dua konverter I/V yaitu IV-1 dan IV-2. Masing-masing terhubung dengan PSAN-1 dan PSAN-2. Masing-masing pin dihubungkan seperti pada Gambar 4.3.

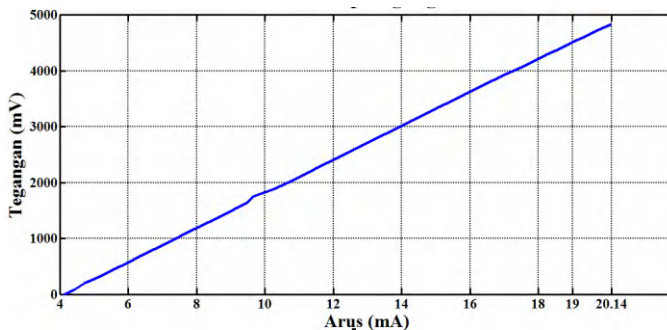


Gambar 4.3 Pin Konfigurasi Pengujian

Pengujian konverter I/V dilakukan dengan cara memvariasi arus masuk 4-20 mA ke rangkaian konverter I/V. Agar arus dapat divariasikan, maka *pressure* (tekanan) yang masuk ke PSAN harus divariasikan pula dengan cara mengubah proses *flow/level*. Hasil dari pengujian seperti pada Gambar 4.4 dan Gambar 4.5.



Gambar 4.4 Fungsi Tegangan terhadap Arus IV-1



Gambar 4.5 Fungsi Tegangan terhadap Arus IV-2

Dari kedua Gambar 4.4 dan Gambar 4.5 dapat disimpulkan bahwa rangkaian yang dihasilkan adalah linier/proporsional. Sesuai dengan Persamaan (4.1):

$$V = 0,3125I - 1,25 \quad (4.1)$$

4.1.2.1 Pengujian V/I Converter (VI-1)

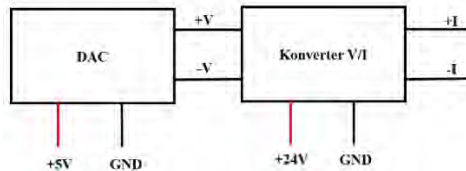
Pengujian kali ini dimaksudkan untuk mengetahui tingkat kelinieran Konverter V/I. Pengujian konverter V/I dilakukan dengan cara memvariasi tegangan input 0-5 V yang masuk ke rangkaian konverter V/I.

Sebelum melakukan pengujian, alat dikalibarsi terlebih dahulu. Kalibrasi pada VI-1 dilakukan pada *multiturn* R1 dan R2. R1 dan R2 di atur untuk menentukan batas bawah dan batas atas dari VI-1.

Penentuan batas bawah dilakukan dengan memberikan tegangan 0 volt pada input VI-1. Kemudian ukur arus *output* VI-1 menggunakan amperemeter. Lalu, arus output VI-1 diatur hingga menunjukkan nilai 4 mA dengan cara memutar knob *multiturn* R1.

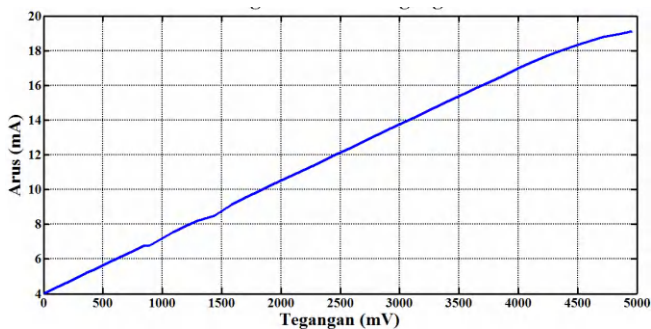
Penentuan batas atas dilakukan dengan memberikan tegangan 5 Volt pada input VI-1. Kemudian ukur arus output VI-1 menggunakan amperemeter. Lalu, arus output VI-1 diatur hingga menunjukkan nilai 20 mA dengan cara memutar knob *multiturn* R2.

Dalam pengambilan data, tegangan 0-5 V sebagai masukan dari VI-1 didapat dari *output Digital to Converter* (DAC) dengan cara memberi masukan bit DAC 0-4095 seperti Gambar 4.6.



Gambar 4.6 Konfigurasi Pengujian VI-1

Dari hasil pengukuran, hubungan antara tegangan dan arus yang dihasilkan rangkaian V/I dapat dilihat pada Gambar 4.7



Gambar 4.7 Fungsi Arus terhadap Tegangan VI-1

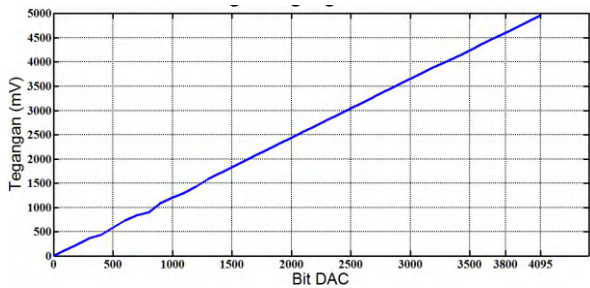
Dari Gambar 4.7, dapat terlihat bahwa hubungan antara nilai tegangan dan arus menghasilkan grafik yang linier. Hubungan antara tegangan dan arus ini dapat ditulis dengan Persamaan (4.2).

$$I = 3,2 V + 4 \quad (4.2)$$

4.1.2 Pengujian DAC

Pengujian DAC dilakukan dengan cara memberi bit 0-4095 pada M-DAQ. Lalu tegangan terukur diambil dari pin 14 DAC (*channel 0*).

Pemberian bit pada DAC menggunakan program di LabVIEW. Hasil dari pengukuran dapat dilihat pada Gambar 4.8.

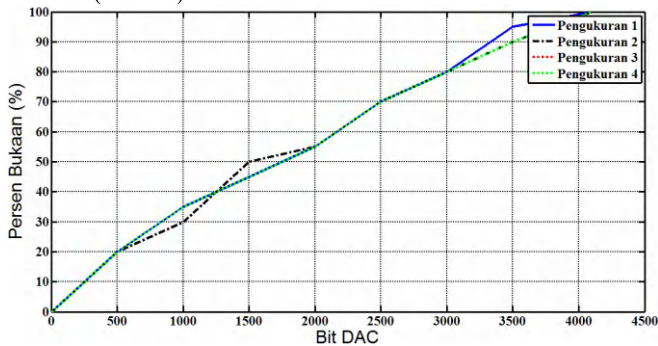


Gambar 4.8 Fungsi Bit DAC terhadap Tegangan Output

Dari Gambar 4.8 terlihat bahwa hubungan antara tegangan dan bit DAC sangat linier. Hal ini dikarenakan, komponen tersebut mempunyai resolusi yang cukup tinggi yaitu 12 bit. Artinya, setiap satu bit merepresentasikan 1.22 mV keluaran tegangan DAC untuk V_{REF} 5 V. DAC inilah yang akan bertugas untuk memberikan sinyal kontrol yang dihasilkan oleh kontroler digital di komputer dengan *plant* Yokogawa Tipe S4.

4.1.3 Pengujian *Control Valve* (PCV-1)

Pengujian ini dilakukan untuk mengetahui karakteristik dari *control valve* (PCV-1).



Gambar 4.9 Fungsi Buka Valve terhadap Bit DAC

PCV-1 tersebut merupakan aktuator *plant* dengan maksimum ketinggian bukaan valve 10 mm. Untuk pengujiannya, *control valve* digerakan melalui konverter I/P. Pengujian dilakukan dengan cara memberikan arus pada I/P. Sesuai dengan pengujian yang telah dilakukan pada DAC dan konverter V/I, maka untuk memanipulasi arus masukan (4-20) ke I/P, cukup dikendalikan melalui komputer dengan cara memberikan bit pada DAC. Hubungan antara bit DAC dengan bukaan valve dalam hal ini adalah ketinggiannya, disajikan pada Gambar 4.9.

Dari Gambar 4.9, dapat disimpulkan bahwa karakteristik dari control valve (PCV-1) adalah linier dengan kondisi *air to open* (AC). Artinya, *valve* akan membuka apabila diberi tekanan udara.

4.1.4 Pengujian Sensor Model Plant Yokogawa Tipe S4

4.1.4.1 Sensor Level (LT-1)

Sesuai dengan penjelasan pada Subbab 2.1, keluaran dari LT-1 berupa *pressure* (tekanan). Sehingga diperlukan sensor *pressure* Autonics (PSAN)-2 untuk mengukur tekanan LT-1. Sensor Autonics tersebut sebagai pengganti indikator analog dari LT-1 sekaligus konverter sinyal pneumatik ke listrik agar dapat diolah oleh komputer. Dalam pengujiannya, keluaran dari LT-1 dibaca melalui indikator PSAN. Sementara itu, masukan dari LT-1 dapat diamati melalui LR-1 yang berada dipanel lama dengan nilai 0-50 cm.

Cara pengujiannya adalah dengan cara memanipulasi bukaan *control valve* (PCV-1) sampai *level* mencapai nilai yang telah ditentukan sebelumnya. Dari kurva pada Gambar 4. 10, dapat dilihat bahwa prosentase bacaan ADC M-DAQ tidak sesuai dengan prosentase *level*. Untuk menyesuaikan prosentase hasil baca ADC M-DAQ agar sesuai dengan prosentase *level*, maka dilakukan regresi. Regresi ini dilakukan dengan *software* LabVIEW.

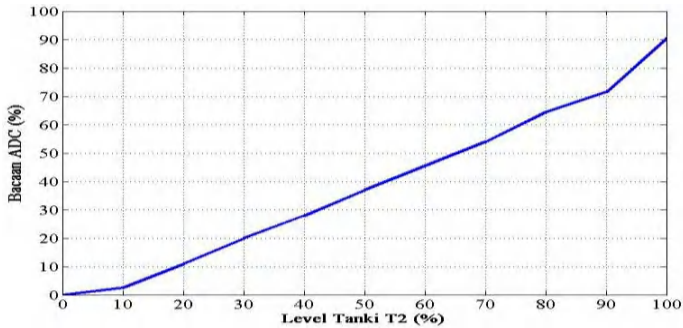
Ada tiga buah macam bentuk fungsi yang digunakan untuk mendekati pola kurva bacaan ADC M-DAQ terhadap *level*, yaitu fungsi linier, fungsi kuadrat, dan fungsi akar. Fungsi pendekatan hasil regresi linier (\hat{y}_1), kuadrat (\hat{y}_2), dan akar (\hat{y}_3) ini dapat dilihat pada Persamaan (4.3), (4.4), dan (4.5).

$$\hat{y}_a = 1,16x + 4,54 \quad (4.3)$$

$$\hat{y}_b = (0,11x + 1,84)^2 \quad (4.4)$$

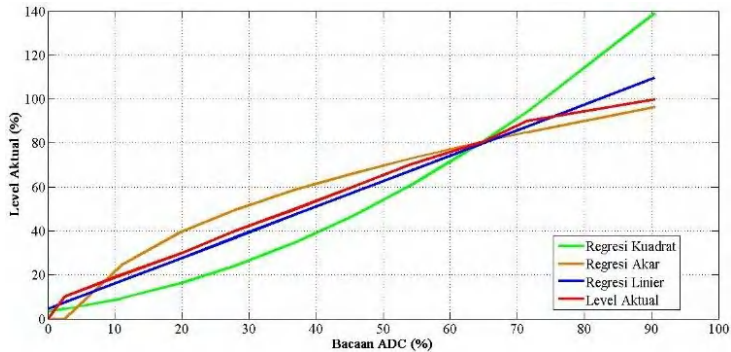
$$\hat{y}_c = \sqrt{109,24x + 601,42} \quad (4.5)$$

Hasil pengujian dapat dilihat pada Gambar 4. 10.



Gambar 4. 10 Hasil Bacaan Level LT-1

Hasil dapat dilihat pada Gambar 4.11.



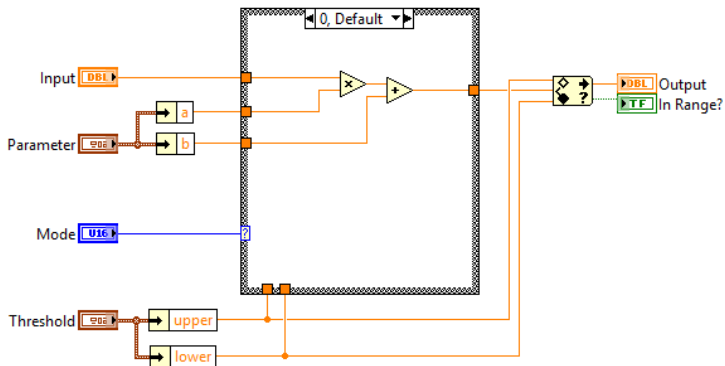
Gambar 4.11 Regresi Bacaan Level

Dari kurva Gambar 4.11 tersebut dapat disimpulkan bahwa hasil regresi fungsi linier sangat mendekati nilai *level* aktual. Sehingga fungsi (\hat{y}_1) nantinya akan digunakan pada *software* LabVIEW. Hasil nilai RMSE dari masing-masing pendekatan regresi dapat dilihat pada Tabel 4. 1.

Tabel 4. 1 Hasil Regresi pada *Level*

Pendekatan Regresi	RMSE
Regresi Linier	3,9013
Regresi Kuadrat	15,4648
Regresi Akar	6,5148

Dari Tabel 4. 1, diperoleh hasil bahwa regresi bentuk linier memiliki RMSE paling kecil. Dengan demikian, hasil konversi data dari M-DAQ terlebih dahulu dilakukan regresi dengan membuat subVI pada LabVIEW khusus untuk regresi linier. Realisasi subVI untuk regresi akar ini dapat dilihat pada Gambar 4.12.



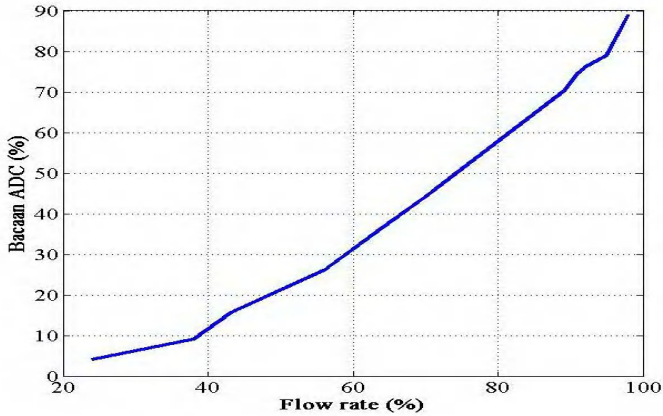
Gambar 4.12 Algoritma LabVIEW untuk Regresi Linier

4.1.4.2 Sensor Flow (FT-1)

Sesuai dengan penjelasan pada Subbab 2.1, keluaran dari FT-1 berupa *pressure* (tekanan). Sehingga diperlukan sensor *pressure* Autonics (PSAN)-1 untuk mengukur tekanan FT-1. Sensor Autonics tersebut sebagai pengganti indikator analog dari FT-1 sekaligus konverter sinyal pneumatik ke listrik agar dapat diolah oleh komputer. Dalam pengujiannya, keluaran dari FT-1 dibaca melalui indikator PSAN. Sementara itu, masukan dari FT-1 dapat diamati melalui FR-1 yang berada dipanel lama sekaligus untuk kalibrasi sensor PSAN-1.

Cara pengujiannya adalah dengan cara memanipulasi bukaan *control valve* (PCV-1) sampai *flow* mencapai nilai yang telah ditentukan sebelumnya. Hasil baca ADC yang memiliki range antara 0-4095 dirubah ke dalam persen, yaitu 0-100%. Sedangkan nilai *flow rate* dengan range

0-500 l/h juga diubah kedalam persen, yaitu 0-100%. Hasil pembacaan *flow* oleh LabVIEW dapat dilihat pada Gambar 4.13.



Gambar 4.13 Hasil Bacaan Flow FT-1

Dari kurva Gambar 4.13 diatas, dapat dilihat bahwa prosentase bacaan ADC M-DAQ tidak sesuai dengan prosentase *flow rate*. Untuk menyesuaikan prosentase hasil baca ADC M-DAQ agar sesuai dengan prosentase *flow rate*, maka dilakukan regresi. Regresi ini dilakukan dengan *software* LabVIEW.

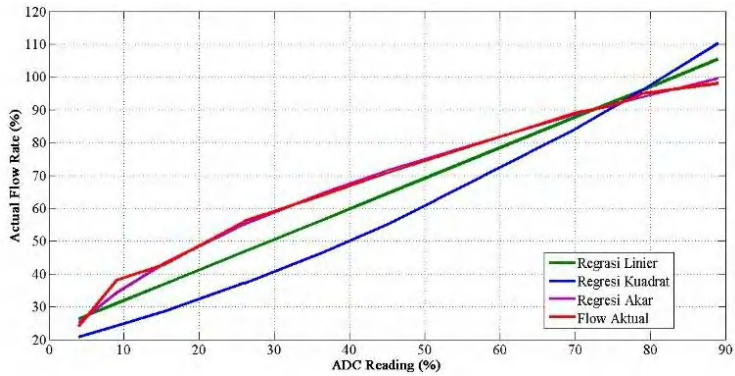
Ada tiga buah macam bentuk fungsi yang digunakan untuk mendekati pola kurva bacaan ADC M-DAQ terhadap *flow rate*, yaitu fungsi linier, fungsi kuadrat, dan fungsi akar. Fungsi pendekatan hasil regresi linier (\hat{y}_a), kuadrat (\hat{y}_b), dan akar (\hat{y}_c) ini dapat dilihat pada Persamaan (4.6), (4.7), dan (4.8)

$$\hat{y}_a = 0,93x + 22,59 \quad (4.6)$$

$$\hat{y}_b = (0,07x + 4,27)^2 \quad (4.7)$$

$$\hat{y}_c = \sqrt{109,08x + 191,95} \quad (4.8)$$

Hasil dari regresi ini dapat dilihat pada Gambar 4.14.



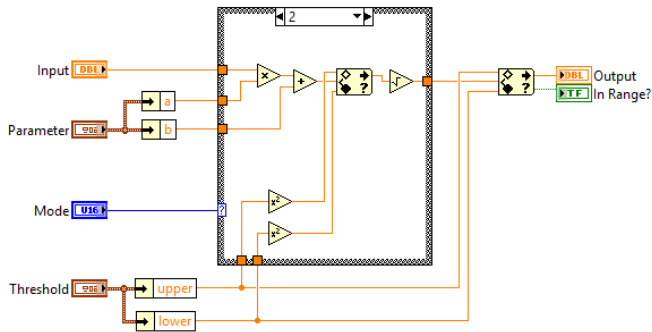
Gambar 4.14 Regresi Bacaan *Flow*

Dari kurva Gambar 4.14. tersebut dapat disimpulkan bahwa hasil regresi fungsi akar sangat mendekati nilai *flow* aktual. Sehingga fungsi (\hat{y}_c) nantinya akan digunakan pada *software* LabVIEW. Hasil nilai RMSE dari masing-masing pendekatan regresi dapat dilihat pada Tabel 4.2.

Tabel 4.2 Hasil Regresi pada Level

Pendekatan Regresi	RMSE
Regresi Linier	3,9013
Regresi Kuadrat	15,4648
Regresi Akar	6,5148

Dari Tabel 4.2, diperoleh hasil bahwa regresi bentuk akar memiliki RMSE paling kecil. Dengan demikian, hasil konversi data dari M-DAQ terlebih dahulu dilakukan regresi dengan membuat subVI pada LabVIEW khusus untuk regresi akar. Realisasi subVI untuk regresi akar ini dapat dilihat pada Gambar 4.15.



Gambar 4.15 Algoritma LabVIEW untuk Regresi Akar

4.2 Pengujian Simulasi

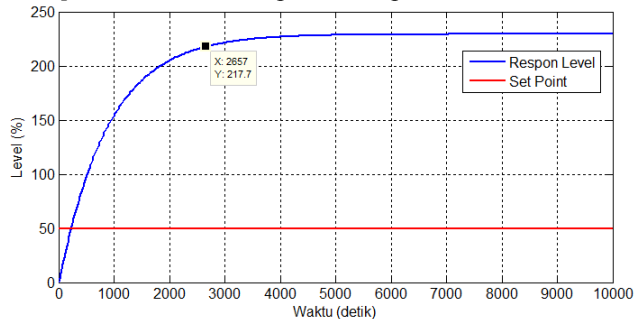
Tujuan dari simulasi adalah menemukan PID Prediktif terbaik dengan variasi nilai variabel tuning rw yang telah dihitung pada Tabel 3.10. Hasil terbaik dari simulasi akan dibuat implementasi. Simulasi kontroler dilakukan menggunakan NI LabVIEW 2013. *Front panel* dari simulator dapat dilihat pada Gambar 4. 16. Pada *front panel* tersebut terdapat *dialog box* untuk memilih metode pembebanan yaitu nominal, minimal, dan maksimal. Untuk parameter kontroler pada Tabel 3.10, dapat dimasukan pada *front panel* tersebut. Untuk diagram blok simulasi dapat dilihat pada Lampiran.



Gambar 4. 16 *Front Panel* Simulasi

4.2.1 Pengujianji Respon Tanpa Kontroler

Respon plant level tanpa kontroler pada beban nominal dengan dengan *set point* 50 % level dapat dilihat pada Gambar 4. 17.

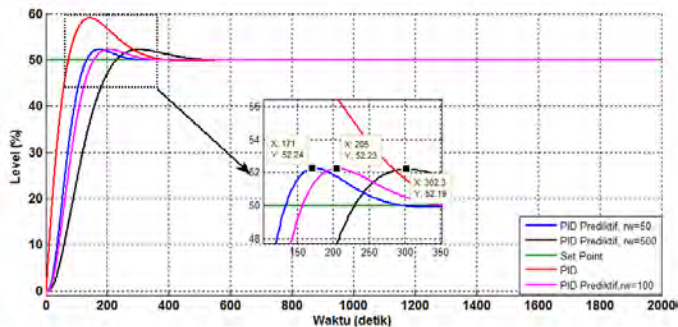


Gambar 4. 17 Respon Level Tanpa Kontroler pada Beban Nominal

Dari Gambar 4. 17 terlihat bahwa respon *level* mencapai *settling time* $ts(\pm 5\%)$ saat 2.657 detik. Dengan $Y_{ss} = 229,24\%$. Dari respon *open loop* tersebut, akan ditambahkan kontroler untuk memperbaiki respon tersebut.

4.2.2. Pengujian Efek Variabel Tuning (*rw*) Terhadap Respon *Level*

Tujuan dari pengujian simulasi adalah menemukan kontroler PID Prediktif terbaik berdasarkan variasi nilai variabel *tuning* (*rw*) dengan jumlah prediksi yang sudah ditentukan sebelumnya, yaitu 200. Hasil perbandingan respon masing-masing variabel *tuning* seperti dapat dilihat pada Gambar 4.18



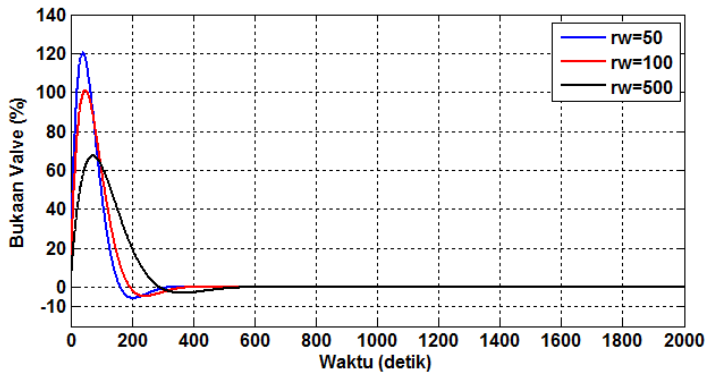
Gambar 4.18 Perbandingan Respon dengan Berbagai Kontroler

Kontroler PID Prediktif terbaik ini dinilai dari nilai *error steady state* terkecil saat simulasi. Pengujian simulasi ini dilakukan dengan memberikan sinyal uji *step*. Masing-masing *gain* dari PID Prediktif dapat dilihat pada Tabel 4.3.

Tabel 4.3 Spesifikasi untuk Setiap Kontroler

Jenis Kontroler	M_p (%)	t_s (5%) (detik)	t_r (5-95%) (detik)	e_{ss} (%)
PID Prediktif, $rw=50$	4,48	319	298	0
PID Prediktif, $rw=100$	4,46	340	316	0
PID Prediktif, $rw=500$	4,38	404	372	0

Dari Gambar 4.18, terlihat bahwa kontroler PID Prediktif mempunyai respon terbaik adalah saat rw sama dengan 500. Karena pada kondisi tersebut, respon *level* mempunyai *overshoot* lebih kecil dibandingkan respon *level* dari kontroler lain dengan M_p (%) = 4,38% dan e_{ss} (%)=0. Selain itu, sinyal kontrol yang dihasilkan juga kurang dari 100 % bukaan *valve*, yaitu hanya 70 % bukaan *valve* seperti terlihat pada Gambar 4.19.



Gambar 4.19 Perbandingan Sinyal Kontrol $rw=50$, $rw=100$, dan $rw=500$

Berbeda halnya dengan PID Prediktif dengan $rw=50$ dan $rw=100$. Kedua mempunyai sinyal kontrol yang melebihi 100%. Secara berturut-turut yaitu 120% dan 103%. Sinyal kontrol yang melebihi 100% akan terpotong karena *control valve* hanya mempunyai *range* 0-100%. Melalui Gambar 4.19, juga dapat disimpulkan, semakin kecil nilai variabel *tuning*

(rw), maka respon akan semakin cepat. Akan tetapi nilai *overshoot maksimum* akan bertambah besar. Hal ini dikarenakan semakin kecil nilai rw , maka *gain* dari PID Prediktif akan semakin besar seperti pada Tabel 3.10.

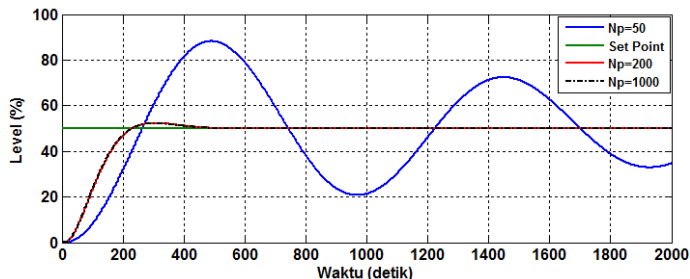
4.2.3 Pengujian Efek Horizon Prediksi (N_p) Terhadap Respon Level

Pengujian ini dilakukan untuk mengetahui pengaruh horizon prediksi terhadap respon *level*. Dalam pengujian, besar nilai N_p divariasikan yaitu 50, 200, dan 1000. Saat $N_p=50$ terlihat bahwa respon membutuhkan waktu *settling* yang lama, yaitu lebih dari 2000 detik. Sementara untuk $N_p=200$ dan 1000 menunjukkan respon yang hampir sama, yaitu waktu *settling*-nya 404. Dari Gambar 4.20 dapat disimpulkan bahwa semakin besar nilai N_p , maka respon yang dihasilkan akan semakin cepat.

Hal ini juga dapat dilihat pada Tabel 4.4. Semakin dekat horizon prediksi dengan waktu *settling open loop level* $t_s(\pm 5\%)$ saat 2.657 detik, respon yang dihasilkan lebih cepat mencapai keadaan tunak (*steady state*). Hal ini dikarenakan derajat kebebasan dari prediksi lebih besar.

Tabel 4.4 Perbandingan Spesifikasi dari Horizon Prediksi

Parameter (N_p)	$M_p(\%)$	$t_s(5\%)$ (detik)
$N_p=50$	79,82	>2000
$N_p=200$	4,46	404
$N_p=1000$	4,46	404

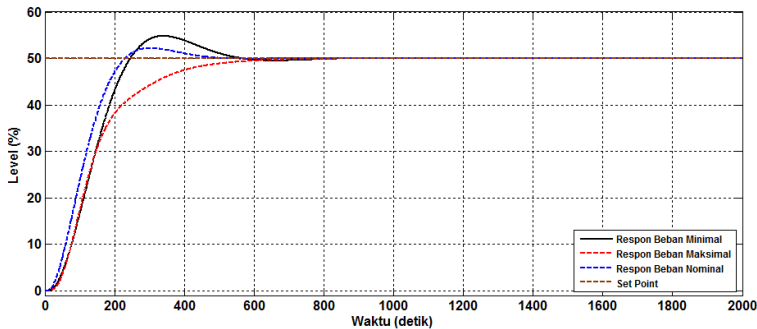


Gambar 4.20 Perbandingan Respon Level dengan $N_p=50$, $N_p=200$, dan $N_p=1000$

4.2.4 Pengujian Kontroler dengan Beban

Pengujian dilakukan pada masing-masing beban dengan parameter yang sama yaitu parameter kontroler pada beban nominal. Dengan nilai masing-masing *gain* seperti pada Tabel 3.10.

$$q_{ff} = 0,044; q_0 = 37,586; q_1 = -70,50; q_2 = 32,96; q_u = 0,061$$



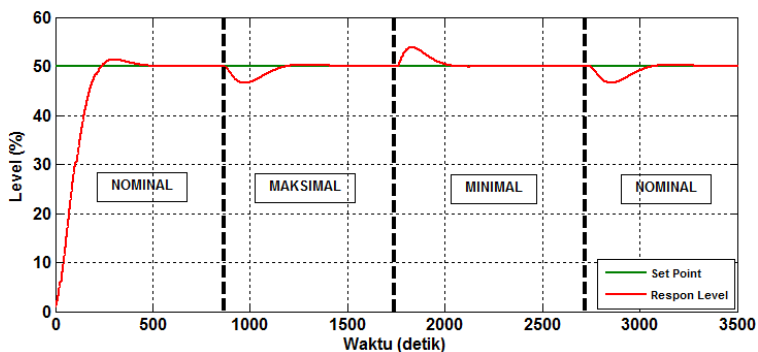
Spesifikasi respon pada masing-masing beban dapat dilihat pada Tabel 4.5 berikut.

Tabel 4.5 Spesifikasi Peforma Kontroler PID Prediktif pada Berbagai Kondisi

Beban	t_s (5%) (detik)	e_{ss} (%)	M_p (%)
Nominal	404	0	4,46
Minimal	550	0	9,60
Maksimal	660	0	0,00

4.2.5 Pengujian Kontroler pada Perubahan Beban

Pengujian simulasi dilakukan dengan cara memberikan sinyal uji *step*. Selain itu, kontroler hasil desain juga diuji coba pada berbagai kondisi beban. Kontroler yang digunakan adalah kontroler yang didesain pada beban nominal. Melalui uji coba tersebut, nantinya dapat diambil kesimpulan mengenai peforma kontroler untuk beban yang berbeda dengan parameter kontroler yang tetap. Pengujian perubahan beban dilakukan saat respon mencapai keadaan tunak (*steady state*)



Gambar 4.21 Respon *Level* dengan Berbagai Transisi Beban pada Simulasi

Tabel 4.6 Spesifikasi Praktis Respon *Level* pada Simulasi dengan Tiga Macam Pembebanan

No.	Kondisi Awal	Kondisi Perubahan	Settling time (5%)	Mp (%)	e_{ss} (%)
1.	0	Nominal	404	4,38	0
2.	Nominal	Maksimal	64	-6,56	0
3.	Maksimal	Minimal	0	10,22	0
4.	Minimal	Nominal	67	-6,80	0

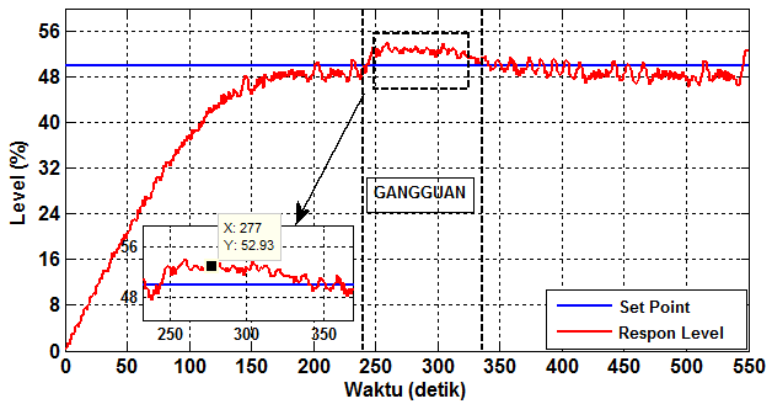
Melalui Tabel 4.6, dapat disimpulkan bahwa pada simulasi, baik pada beban nominal, maksimal, dan minimal tidak ada *error steady state* yang muncul. Namun pada transisi beban dari maksimal ke minimal muncul *overshoot* cukup besar yaitu 10,22%. Dari Tabel 4.6 juga dapat disimpulkan bahwa kontroler mampu bekerja dengan baik meskipun terjadi perubahan beban. Kontroler mampu mengembalikan respon ke keadaan *steady state* dalam waktu 67 detik.

4.3 Pengujian Implementasi

Pengujian implementasi dilakukan setelah melakukan pengujian simulasi. Pengujian implementasi ini dilakukan untuk mengukur performa kontroler saat diterapkan pada *rill plant*. Pengujian implementasi ini terdiri dari tiga pengujian yaitu pengujian respon *level* terhadap gangguan, pengujian respon dengan perubahan *set point*, dan pengujian respon dengan perubahan beban.

4.3.1 Pengujian Kontroler terhadap Gangguan

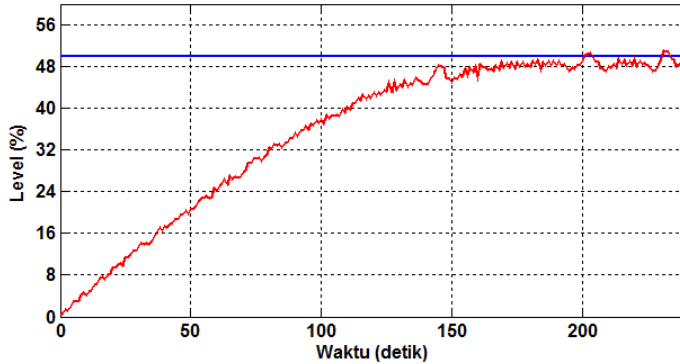
Pengujian dilakukan pada beban nominal yaitu katup beban *Needle Valve* (NV-2) dibuka 5 putaran dari *full close*. Dari Gambar 4.22, terlihat bahwa respon *level* dapat mengikuti *set point* yang diberikan. Pada detik ke-250 diberikan gangguan sesaat pada *flow in*. Pemberian gangguan dilakukan dengan cara membuka F(L)CV-1 dari keadaan 75% bukaan menjadi 100% bukaan untuk beberapa saat kemudian dikembalikan pada posisi semula. Hasil respon akibat pemberian gangguan dapat dilihat pada Gambar 4.22.



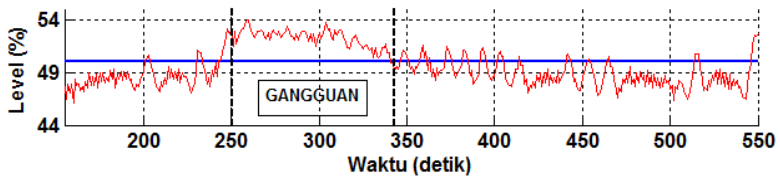
Gambar 4.22 Respon *Level* dengan Gangguan

Pada keadaan transien seperti Gambar 4.23, respon mempunyai waktu *settling* ($\pm 5\%$) pada $t=174$ detik. Sementara t_r (5%-95%) sebesar 169 detik. Jika ditelaah lebih lanjut, respon masih mempunyai *error steady state* (e_{ss}) sebesar 2,98%.

Pemberian gangguan dilakukan saat respon mencapai *steady state* pada detik ke-250. Akibatnya, respon mengalami kenaikan sebesar 2,98 %. Namun pada detik ke-340, *level* mampu kembali pada keadaan tunak (*steady state*).

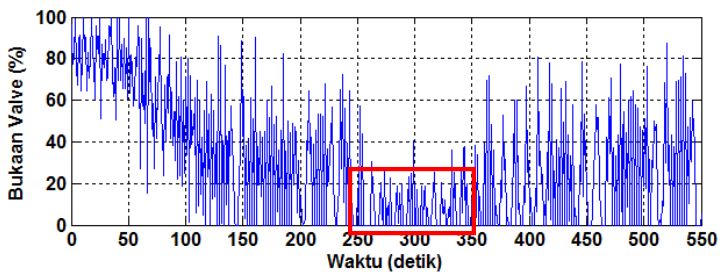


Gambar 4.23 Respon Transien *Level*



Gambar 4.24 Pemberian Gangguan Sesaat pada Valve F(L)CV-1

Menilik pada Gambar 4.24, gangguan sesaat pada *flow* masuk mengakibatkan *level* pada tangki dua (T2) mengalami kenaikan sebesar 1,85 cm (2,98%). Meskipun terjadi kenaikan, kontroler masih mampu mengembalikan respon pada *steady state* dengan waktu 90 detik.

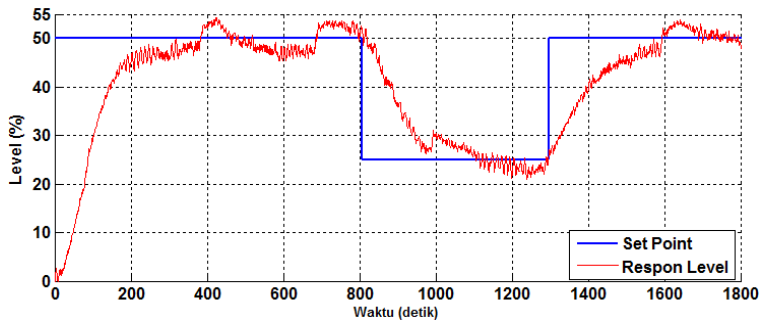


Gambar 4.25 Sinyal Kontrol Akibat Pemberian Gangguan

Terlihat pada Gambar 4.25, untuk mengkompensasi gangguan tersebut, sinyal kontrol yang diberikan pada *control valve* (PCV-1) mengalami penurunan hingga detik ke 340. Setelah 340 detik, sinyal kontrol kembali normal.

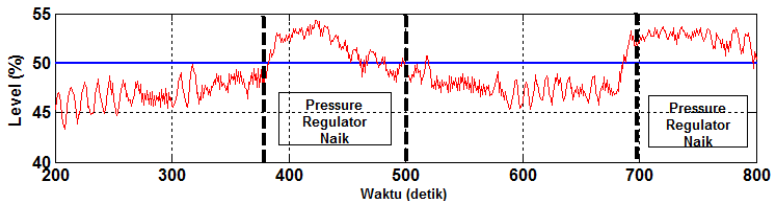
4.3.2 Pengujian Kontroler terhadap Perubahan *Set Point*

Pada Gambar 4.26 terlihat bahwa respon *level* dapat mengikuti perubahan *set point* yang diberikan. *Set point* diberikan pada tiga kondisi yaitu 50%, 25%, dan 50%. Saat pengujian masih terdapat *error steady state* sekitar 4,568%.



Gambar 4.26 Respon *Level* dengan Perubahan *Set Point*

Pada detik ke-375 dan detik ke-675 terjadi *overshoot* berturut-turut sebesar 6,982% dan 6,38%. *Overshoot* tersebut diakibatkan *pressure regulator* yang mengalami kenaikan sehingga F(L)C-1 ikut naik pula seperti pada Gambar 4.27.

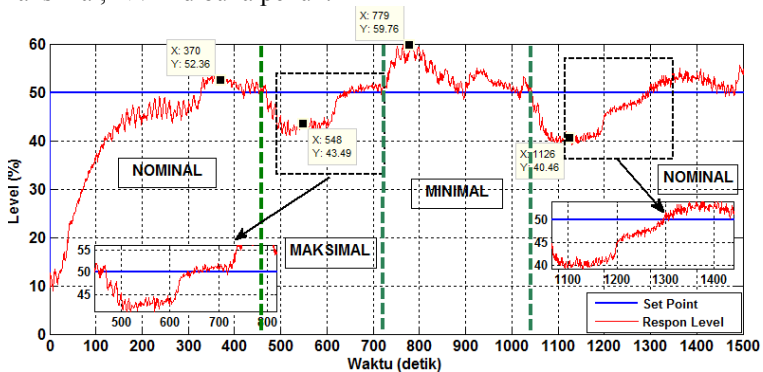


Gambar 4.27 *Overshoot* Akibat *Pressure Regulator Naik*

Namun kontroler masih mampu mengkompensasi gangguan tersebut. Dan mengembalikan respon *level* kembali ke keadaan tunak (*steady state*).

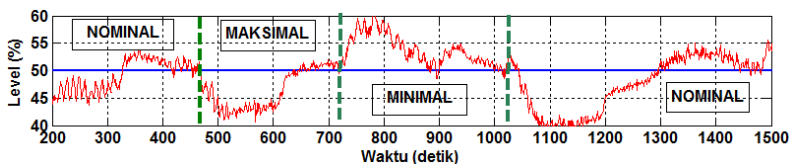
4.3.3 Pengujian Kontroler terhadap Perubahan Beban

Respon *level* hasil implementasi dengan berbagai transisi beban dapat dilihat pada Gambar 4.28. Pemberian beban dilakukan secara kontinu bukan sesaat. Perubahan beban dilakukan melalui katup beban *Needle Valve* (NV-2). Pemberian beban nominal dilakukan dengan cara memutar NV-2 sebanyak 5 putaran dari keadaan *full close*. Sementara beban minimal diputar sebanyak 3 putaran. Terakhir, untuk beban maksimal, NV-2 dibuka penuh.



Gambar 4.28 Respon *Level* dengan Berbagai Transisi Beban pada Implementasi

Dalam pegujian perubahan beban. Beban diberikan saat respon telah mencapai keadaan tunak (*steady state*). Hasil respon saat terjadi perubahan beban dapat dilihat pada Gambar 4.29.



Gambar 4.29 Perubahan Beban Mengakibatkan Respon *Level* Berubah

Dari hasil implementasi pada Gambar 4.28 didapat respon *steady state level* sebagai berikut :

1. Beban Nominal Pertama : $Y_{ss} = 51,784 \%$
2. Beban Maksimal : $Y_{ss} = 50,621 \%$
3. Beban Minimal : $Y_{ss} = 52,131 \%$
4. Beban Nominal Kedua : $Y_{ss} = 51,777 \%$

Merujuk pada Gambar 4.28 dan Gambar 4.29, terlihat bahwa saat transisi dari beban yang lebih besar ke kecil (Maksimal-Minimal) terjadi kenaikan *level* air sebesar 9,78 % atau sekitar 4,89 cm. Sementara untuk perubahan transisi beban kecil ke besar (Nominal-Maksimal/Minimal-Nominal) terjadi penurunan *level* air sebesar 10,46 % atau sekitar 5,23 cm. Namun, kontroler mampu mengembalikan respon menuju *steady state*.

Dari data tersebut dapat diketahui bahwa kontroler mampu mengatasi sistem yang diberikan perubahan beban. Akan tetapi, akibat perubahan beban tersebut membuat respon level turun/naik dari keadaan *steady state* Sedangkan untuk perbandingan, *settling time* (5%), *overshoot maximum* (Mp) dan *error steady state* (e_{ss}) untuk masing-masing transisi pembebanan ditunjukkan pada Tabel 4.7.

Tabel 4.7 Spesifikasi Praktis Respon Level pada Implementasi dengan Tiga Macam Pembebanan

No.	Kondisi Awal	Kondisi Perubahan	<i>Settling time</i> (5%)	Mp (%)	e_{ss} (%)
1.	0	Nominal	206	4,72	3,570
2.	Nominal	Maksimal	59	0,00	1,242
3.	Maksimal	Minimal	0	19,52	4,262
4.	Minimal	Nominal	66	4,61	3,554

BAB V

PENUTUP

5.1 Kesimpulan

Dari hasil analisa terhadap simulasi dan implementasi yang telah dilakukan pada pengerjaan Tugas Akhir ini, diperoleh beberapa kesimpulan:

1. Pada simulasi, kontroler PID Prediktif yang dirancang mempunyai *error steady state* sebesar 0 % pada beban nominal, beban minimal, dan beban maksimal.
2. Pada implementasi, saat terjadi pembebanan pada katup beban, terjadi *error steady state* sebesar 3,57% pada beban nominal, 1,242% pada beban maksimal, dan 4,262% pada beban minimal.
3. Pada pengujian perubahan *set point*, respon *level* mampu mengikuti perubahan *set point* namun masih terdapat *error steady state* sebesar 4,568%.
4. Akibat pemberian gangguan pada *flow* masuk secara sesaat, mengakibatkan kenaikan level sebesar 2,98% atau 1,85 cm. Pemberian gangguan dilakukan dengan cara membuka F(L)CV-1 dari 75% ke 100%. Namun, kontroler mampu mengembalikan respon pada *steady state* selama 90 detik.
5. Semakin kecil *variabel tuning* (rw), maka nilai gain PID Prediktif akan semakin besar pula. Sehingga nilai kontrol yang dihasilkan akan besar pula.
6. Horizon prediksi dipilih berdasarkan waktu respon mencapai 63,2 % dari *steady state*.

5.2 Saran

Pada penelitian selanjutnya, perlu ada perbaikan pada regulator tekanan pneumatik. Hal ini dikarenakan tekanan yang masuk ke *control valve* akan berubah-ubah. sehingga menyebabkan sinyal kontrol yang diberikan pada *control valve* tidak sesuai dengan harapan. Apabila tekanan regulator naik, maka *control valve* akan mengalami perubahan (naik)

-Halaman ini sengaja dikosongkan-

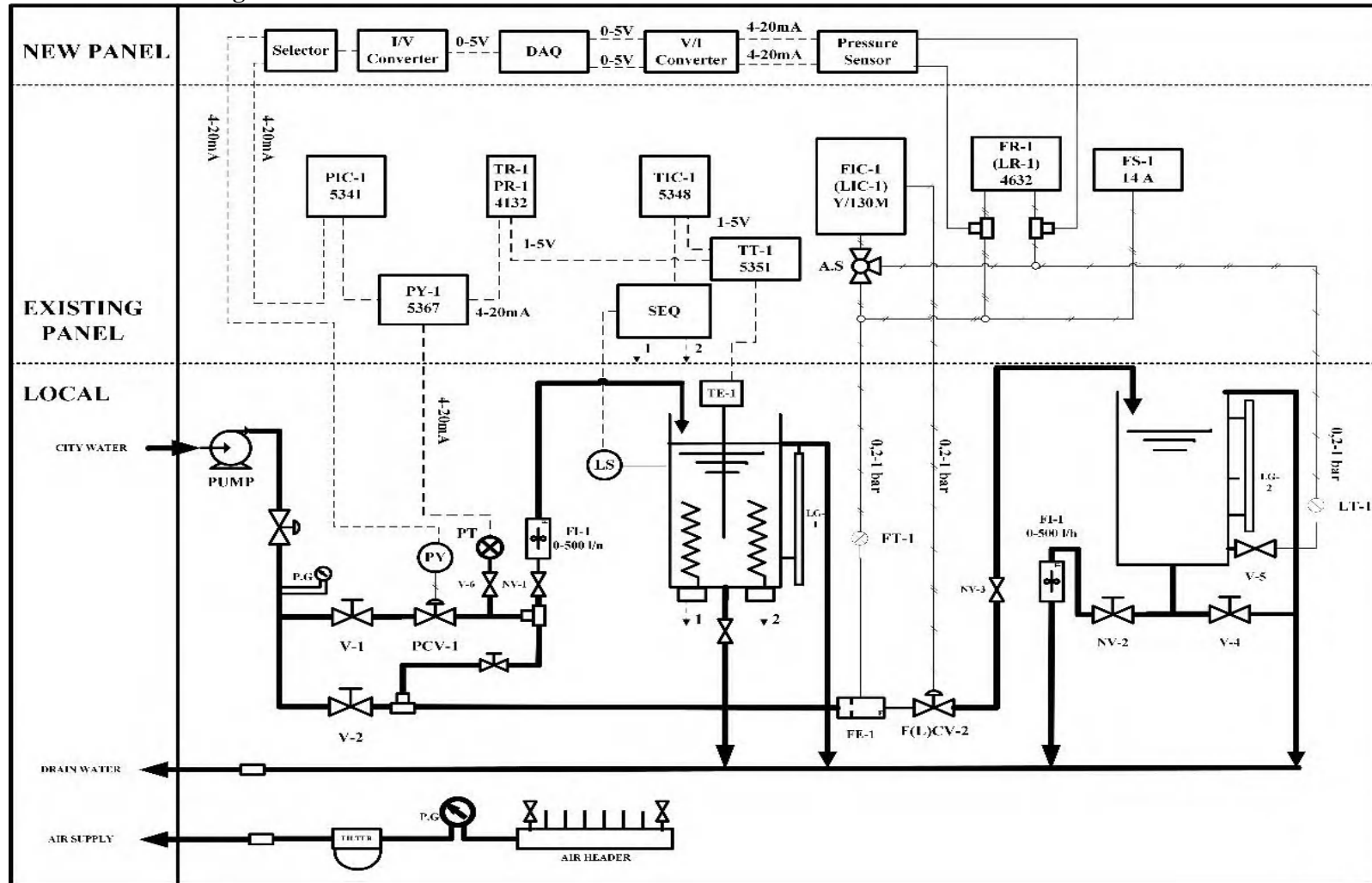
DAFTAR PUSTAKA

- [1] Yokogawa Electric Works, Instruction Manual Type "S-4" Plant Model, Japan, 1982.
- [2] M. G. Na, "Auto-Tuned PID Controller Using a Model Predictive," *IEEE TRANSACTIONS ON NUCLEAR SCIENCE*, vol. 48, p. 5, 2001.
- [3] L. Wang, Model Predictive Control System Design and Implementation Using MATLAB®, London: Springer-Verlag London Limited, 2009.
- [4] M. A. Johnson dan M. H. Moradi, PID Control : New Identification and Design Methods, 2005: Springer-Verlag London , London.
- [5] I. P. Jakoubek, "Experimental Identification of Stable Nonoscillatory Systems from Step-Responses by Selected Methods," dalam *Konference Studentske tvorci cinnosti*, 2009.
- [6] T. A. Wijaya, "Pengaturan Tekanan pada Pressure Process Eig 38-714 melalui ABB Commander 350 Menggunakan Kontroler Fuzzy PID," dalam *Tugas Akhir JTE ITS*, Surabaya, 2011.
- [7] S. A. Nugroho, "Pengendalian Kecepatan Motor Induksi Tiga Fasa Pada Mesin Sentrifugal Dengan Kontroler PID-Predictive Berbasis PLC," dalam *Tugas Akhir JTE*, Surabaya, 2014.
- [8] Burr-Brown, "Precision 4mA to 20mA Current Loop Receiver," Dallas, 1997.
- [9] SHENZHEN SUNYUAN TECHNOLOGY CO., LTD. , "ISO-U(A)PO: DC V/I Signal Isolated Amplifier IC".
- [10] Autonic Corporation, "Digital Pressure Sensor (PSAN Series)," Yongdang-Dong, 2012.
- [11] Arduino, "http://www.arduino.cc," 18 February 2015. [Online]. Available: <http://www.arduino.cc/en/Reference/SPI>. [Diakses 18 February 2015].
- [12] S. Sumanthi, LabVIEW based Advanced Instrumentation Systems, New York: Springer-Verlag Berlin Heidelberg, 2007.
- [13] R. Bitter, LabVIEW Advanced Programming Techniques, New York: CRC, 2001.

- [14] Atmel Corporation , “Atmel 8-bit Microcontroller with 4/8/16/32KBytes,” California, 2013.
- [15] Microchip Technology Inc, “MCP3204/3208 2.7V 4-Channel/8-Channel 12-Bit A/D Converters Datasheet,” 2002.
- [16] Microchip Technology Inc, “MCP4902/4912/4922 8/10/12-Bit Dual Voltage Output Digital-to-Analog Converter Datasheet,” 2010.
- [17] D. Uduehi, A. Ordys dan M. Grimbale, “PREDICTIVE PID CONTROLLER DESIGN,” dalam *Conference on Control Applications* , Mexico City,, 2001.
- [18] K. Uren dan v. S. George, “Predictive PID Control of Non-Minimum Phase Systems,” dalam *Advances in PID Control*, Slavka Krautzeka, InTech, 2011, pp. 1-22.

LAMPIRAN

A. Diagram P&ID Setelah Mengalami Perrubahan



[illegible]

-Halaman ini sengaja dikosongkan-

C. Program Identifikasi pada Arduino

```
//-----  
//                                LIBRARY  
//-----  
#include <SPI.h>  
  
//-----  
//                                PARAMETER  
//-----  
#define ss1 10  
#define ss2 9  
#define ss3 8  
#define ss4 7  
#define runLED A5  
#define di0 A1  
#define di1 A4  
#define di2 A2  
#define di3 A3  
#define di4 A0  
#define do0 6  
#define do1 2  
#define do2 5  
#define do3 4  
#define do4 3  
  
//-----  
//                                GLOBAL VARIABLE  
//-----  
//Declaration of variable that will be used for synchronizing  
byte syncIn[8];  
byte match = 0;  
byte syncByte;  
String r_sync = "GETREADY";  
String s_sync = "LETSROLL";  
//Declaration of variable that will be used for processing command  
byte commandIn[6];  
byte commandOut[6] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00};  
byte commandByte;  
byte crc;
```

```

//Build array for SPI communication with MCP3204
byte adcDataIn[3];
byte adcDataOut[3];
//Build array for SPI communication with MCP4922
byte dacDataOut[2];
//ADC Reading result
word readVal;

```

```

//-----
//                               INITIATION
//-----
void setup() {
    // put your setup code here, to run once:
    //declare status pin
    pinMode(runLED, OUTPUT);
    //declare dedicated output pins
    pinMode(do0, OUTPUT);
    pinMode(do1, OUTPUT);
    pinMode(do2, OUTPUT);
    pinMode(do3, OUTPUT);
    pinMode(do4, OUTPUT);
    //declare dedicated input pins
    pinMode(di0, INPUT);
    pinMode(di1, INPUT);
    pinMode(di2, INPUT);
    pinMode(di3, INPUT);
    pinMode(di4, INPUT);
    //output pins all set LOW
    digitalWrite(do0, LOW);
    digitalWrite(do1, LOW);
    digitalWrite(do2, LOW);
    digitalWrite(do3, LOW);
    digitalWrite(do4, LOW);
    //SPI Initialize
    SPI.setBitOrder(MSBFIRST);           // Data in MSB first
    SPI.setDataMode(SPI_MODE0);         // Set SPI mode [CPOL = 0 ;
    CPHA = 0]
    SPI.setClockDivider(SPI_CLOCK_DIV8); // Clock 2 MHz

```

```

//slave pin
pinMode(ss1, OUTPUT);
pinMode(ss2, OUTPUT);
pinMode(ss3, OUTPUT);
pinMode(ss4, OUTPUT);
//slave idle
digitalWrite(ss1, HIGH);      // Slave 1 idle
digitalWrite(ss2, HIGH);      // Slave 2 idle
digitalWrite(ss3, HIGH);      // Slave 3 idle
digitalWrite(ss4, HIGH);      // Slave 4 idle
SPI.begin();                  // Start SPI
Serial.begin(115200);
//Add additional setup here:
//Remember! this code will be excuted before sync sequence start
for (byte i = 0 ; i < 4 ; i++)
{
    //Notice user when setup has finished
    digitalWrite(runLED, HIGH);
    delay(250);
    digitalWrite(runLED, LOW);
    delay(250);
}
//Start synchronization sequence with LabVIEW
sync();
}

//-----
//                      MAIN PROGRAM
//-----
void loop() {
    // put your main code here, to run repeatedly:
    if (Serial.available())
    {
        digitalWrite(runLED, HIGH);
        processCommand();
    }
    //Add additional code below:
    //Remember! adding additional code will slow down
    //communication speed between DAQ and LabVIEW

```

```

digitalWrite(runLED, LOW);

}

//-----
//                      SUB - PROGRAM
//-----

/**SYNCRONIZE***/
void sync()
{
    //start synchronizing
    start_sync:
    while (Serial.available() < 8)
    {
        //do nothing
        //wait for LabVIEW send all sync command
        //if 8 bytes available at serial buffer, loop will end
    }
    //read data from serial buffer
    for (syncByte = 0 ; syncByte < 8 ; syncByte++)
    {
        syncIn[syncByte] = Serial.read();
    }
    //flush serial buffer
    Serial.flush();
    //matching the received data with r_sync
    for (syncByte = 0 ; syncByte < 8 ; syncByte++)
    {
        if (syncIn[syncByte] == r_sync[syncByte])
        {
            //increment if data match
            match++;
        }
    }
    //if all data match, send the sync response
    if (match == 8)
    {

```

```

    for (syncByte = 0 ; syncByte < 8; syncByte++)
    {
        Serial.print(s_sync[syncByte]);
    }
}
else
{
    //synchronize again
    goto start_sync;
}
}

```

```

/**PROCESS COMMAND FROM LABVIEW**
void processCommand()
{
    //start process command
    while (Serial.available() < 6)
    {
        //do nothing
        //wait for LabVIEW send command
        //if 6 bytes available at serial buffer, loop will end
    }
    //read data from serial buffer
    for (commandByte = 0 ; commandByte < 6 ; commandByte++)
    {
        commandIn[commandByte] = Serial.read();
    }
    //flush serial buffer
    Serial.flush();
    //PROTOCOL STATRT HERE!
    //checking CRC
    crc = 0;
    for (commandByte = 0 ; commandByte < 5 ; commandByte++)
    {
        crc = crc + commandIn[commandByte];
    }
    if (crc != commandIn[5])
    {

```



```

//packet data not valid
//request to retransmit
commandOut[0] = 'N';
commandOut[1] = 'A';
commandOut[2] = 'K';
commandOut[3] = '0';
commandOut[4] = '0';
goto sendResponse;
}
else
{
//packet data valid
commandOut[0] = 'A';
commandOut[1] = 'C';
commandOut[2] = 'K';
//success receive command
//accessing peripheral I/O
switch (commandIn[0])
{
case 0x00:
//Single Ended Analog Input
//read MCP3204 ADC Function
//adcRead(adcSlot, adcMode, adcChannel)
readVal = adcRead(slotNumber(commandIn[1]), 0x00,
commandIn[2]);
//Send adc result in 2 byte
commandOut[3] = highByte(readVal);
commandOut[4] = lowByte(readVal);
break;
case 0x01:
//Pseudo Differential Analog Input
//read MCP3204 ADC Function
//adcRead(adcSlot, adcMode, adcChannel)
readVal = adcRead(slotNumber(commandIn[1]), 0x01,
commandIn[2]);
//Send adc result in 2 byte
commandOut[3] = highByte(readVal);
commandOut[4] = lowByte(readVal);
break;

```

```

case 0x02:
    //Analog Output [Output Gain = 1]
    //write MCP4922 DAC Function
    //dacWrite(dacSlot, dacChannel, dacHighByte, dacLowByte)
    dacWrite(slotNumber(commandIn[1]), 0x00, commandIn[2],
commandIn[3], commandIn[4]);
    //send written value
    commandOut[3] = commandIn[3];
    commandOut[4] = commandIn[4];
    break;
case 0x03:
    //Analog Output [Output Gain = 2]
    //write MCP4922 DAC Function
    //dacWrite(dacSlot, dacChannel, dacHighByte, dacLowByte)
    dacWrite(slotNumber(commandIn[1]), 0x01, commandIn[2],
commandIn[3], commandIn[4]);
    //send written value
    commandOut[3] = commandIn[3];
    commandOut[4] = commandIn[4];
    break;
case 0x04:
    //Digital Output (single)
    //write dedicated CPU Digital Output
    if (commandIn[1] == 0x00)
    {
        //digitalWrite(pin, status)
        digitalWrite(dedicatedDigitalOutput(commandIn[2]),
commandIn[3]);
        //send pin and status written
        commandOut[3] = commandIn[2];
        commandOut[4] = commandIn[3];
    }
    else
    {
        commandOut[3] = 'E';
        commandOut[4] = 'R';
    }
    break;
case 0x05:

```

```

//Digital Output (multi)
//write dedicated CPU Digital Output
if (commandIn[1] == 0x00)
{
    //digitalWriteMulti(pinGroup, statusGroup);
    digitalWriteMulti(commandIn[2], commandIn[3]);
    //send pin and status written
    commandOut[3] = commandIn[2];
    commandOut[4] = commandIn[3];
}
else
{
    commandOut[3] = 'E';
    commandOut[4] = 'R';
}
break;
case 0x06:
    //Digital Input (single)
    //Read dedicated CPU Digital Input
    if (commandIn[1] == 0x00)
    {
        if (digitalRead(dedicatedDigitalInput(commandIn[2])) == HIGH)
        {
            commandOut[3] = commandIn[2];
            commandOut[4] = 0x01;
        }
        else
        {
            commandOut[3] = commandIn[2];
            commandOut[4] = 0x00;
        }
    }
    else
    {
        commandOut[3] = 'E';
        commandOut[4] = 'R';
    }
    break;
case 0x07:

```

```

//Digital Input (multi)
//read dedicated CPU Digital Input
if (commandIn[1] == 0x00)
{
    commandOut[3] = commandIn[1];
    commandOut[4] = digitalReadMulti(commandIn[1]);
}
else
{
    commandOut[3] = 'E';
    commandOut[4] = 'R';
}
break;
default:
//Unkonown command
//NOP
commandOut[3] = 'E';
commandOut[4] = 'R';
break;
}
}
sendResponse:
//generate crc:
crc = 0;
for (commandByte = 0 ; commandByte < 5 ; commandByte++)
{
    crc = crc + commandOut[commandByte];
}
commandOut[5] = crc;
//send_response:
for (commandByte = 0 ; commandByte < 6 ; commandByte++)
{
    //Send the value as ASCII code to LabVIEW
    Serial.print(char(commandOut[commandByte]));
}
}

//***Write DAC Value***

```

```
void dacWrite(int dacSlot, byte dacMode, byte dacChannel, byte
dacHighByte, byte dacLowByte)
```

```
{
    //Construct packet to be sent via SPI
    //Default channel 0 & Gain = 1
    dacDataOut[0] = B01110000;
    //Last packet as DAC Low Byte
    dacDataOut[1] = dacLowByte;
    //Mode selection
    if (dacMode == 1)
    {
        //Change Gain = 2
        bitSet(dacDataOut[0], 5);
    }
    //Channel selection
    if (dacChannel == 1)
    {
        //Change to channel 1
        bitSet(dacDataOut[0], 7);
    }
    bitWrite(dacDataOut[0], 0, bitRead(dacHighByte, 0));
    bitWrite(dacDataOut[0], 1, bitRead(dacHighByte, 1));
    bitWrite(dacDataOut[0], 2, bitRead(dacHighByte, 2));
    bitWrite(dacDataOut[0], 3, bitRead(dacHighByte, 3));
    //Send data to MCP4922
    digitalWrite(dacSlot, LOW);    //Start Talking to MCP4922
    SPI.transfer(dacDataOut[0]);    //Send first packet
    SPI.transfer(dacDataOut[1]);    //Send last packet
    digitalWrite(dacSlot, HIGH);    //Stop talking
}
```

```
/**Read ADC Value**/
```

```
word adcRead(int adcSlot, byte adcMode, byte channel)
```

```
{
    //Default is set to Single-Ended Mode
    adcDataOut[0] = B00000111;
    if (adcMode == 1)
    {
```

```

    //pseudo differential mode
    adcDataOut[0] = B00000101;
}
//Select desired channel
switch(channel)
{
    case 0:
        //Channel 0
        adcDataOut[1] = B00000000;
        break;
    case 1:
        //Channel 1
        adcDataOut[1] = B01000000;
        break;
    case 2:
        //Channel 2
        adcDataOut[1] = B10000000;
        break;
    case 3:
        //Channel 3
        adcDataOut[1] = B11000000;
        break;
}
//Request data to MCP3204 chip
digitalWrite(adcSlot, LOW);           // Start talking
delayMicroseconds(1);                 // Give time to finish sampling
(1.5 clk)
adcDataIn[0] = SPI.transfer(adcDataOut[0]); // Send first packet
adcDataIn[1] = SPI.transfer(adcDataOut[1]); // Send second packet
adcDataIn[2] = SPI.transfer(adcDataOut[2]); // Send last packet
digitalWrite(adcSlot, HIGH);           // Stop talking
//Process data
bitClear(adcDataIn[1], 7);
bitClear(adcDataIn[1], 6);
bitClear(adcDataIn[1], 5);
bitClear(adcDataIn[1], 4);
word adcResult = word(adcDataIn[1], adcDataIn[2]);
return adcResult;
}

```

```

****Digital write multi channel****
void digitalWriteMulti(byte outputPinGroup, byte outputPinStatus)
{
    for (byte outputPinCount = 0 ; outputPinCount < 5 ;
outputPinCount++)
    {
        if (bitRead(outputPinGroup, outputPinCount) == HIGH)
        {
            digitalWrite(dedicatedDigitalOutput(outputPinCount),
bitRead(outputPinStatus, outputPinCount));
        }
    }
}

```

```

****Digital read multi channel****
byte digitalReadMulti(byte inputPinGroup)
{
    byte groupReadResult = 0x00;
    for (byte inputPinCount = 0 ; inputPinCount < 5 ; inputPinCount++)
    {
        if (bitRead(inputPinGroup, inputPinCount) == HIGH)
        {
            bitWrite(groupReadResult, inputPinCount,
digitalRead(dedicatedDigitalInput(inputPinCount)));
        }
    }
}

```

```

****Slot Definition****
int slotNumber(byte slotCommand)
{
    //Define slot depend on what LabVIEW send to DAQ
    switch (slotCommand)
    {
        case 0x01:

```

```

        //If slot 1 is choosen
        return ss1;
        break;
case 0x02:
    //If slot 2 is choosen
    return ss2;
    break;
case 0x03:
    //If slot 3 is choosen
    return ss3;
    break;
case 0x04:
    //If slot 4 is choosen
    return ss4;
    break;
default:
    //Default is set to slot 0 - CPU
    return 0;
}
}

int dedicatedDigitalOutput(byte outputNumber)
{
    switch (outputNumber)
    {
        case 0x00:
            return do0;
            break;
        case 0x01:
            return do1;
            break;
        case 0x02:
            return do2;
            break;
        case 0x03:
            return do3;
            break;
        case 0x04:

```



```
        return do4;
        break;
    }
}
```

```
int dedicatedDigitalInput(byte inputNumber)
{
    switch (inputNumber)
    {
        case 0x00:
            return di0;
            break;
        case 0x01:
            return di1;
            break;
        case 0x02:
            return di2;
            break;
        case 0x03:
            return di3;
            break;
        case 0x04:
            return di4;
            break;
    }
}
```

D. Algoritma PID Prediktif

```
clc;
clear;
a=[1 -1.886 0.886];
b=[-0.001044 0.001629];
A=conv(a,[1 -1]);
Np=50;
rw=0.1;
R=rw*eye(Np,Np);
BarsR=ones(Np,1);

[Ca,Ha]=caha(A,1,Np);
[Cb,Hb]=caha(b,1,Np);

theta=inv(Ca)*Cb;
disp('theta=')
disp(theta)

G=inv(Ca)*Hb;
disp('G=')
disp(G)

F=-inv(Ca)*Ha;
disp('Q=')
disp(F)

K=inv(theta'*theta+R)*theta'
disp(K)
KPID=K*F
RecKPID=KPID(1,:)

Ku=K*G
qu=Ku(1,:)

q0=sum(RecKPID(:,1));
disp('q0=')
disp(q0)
```

```

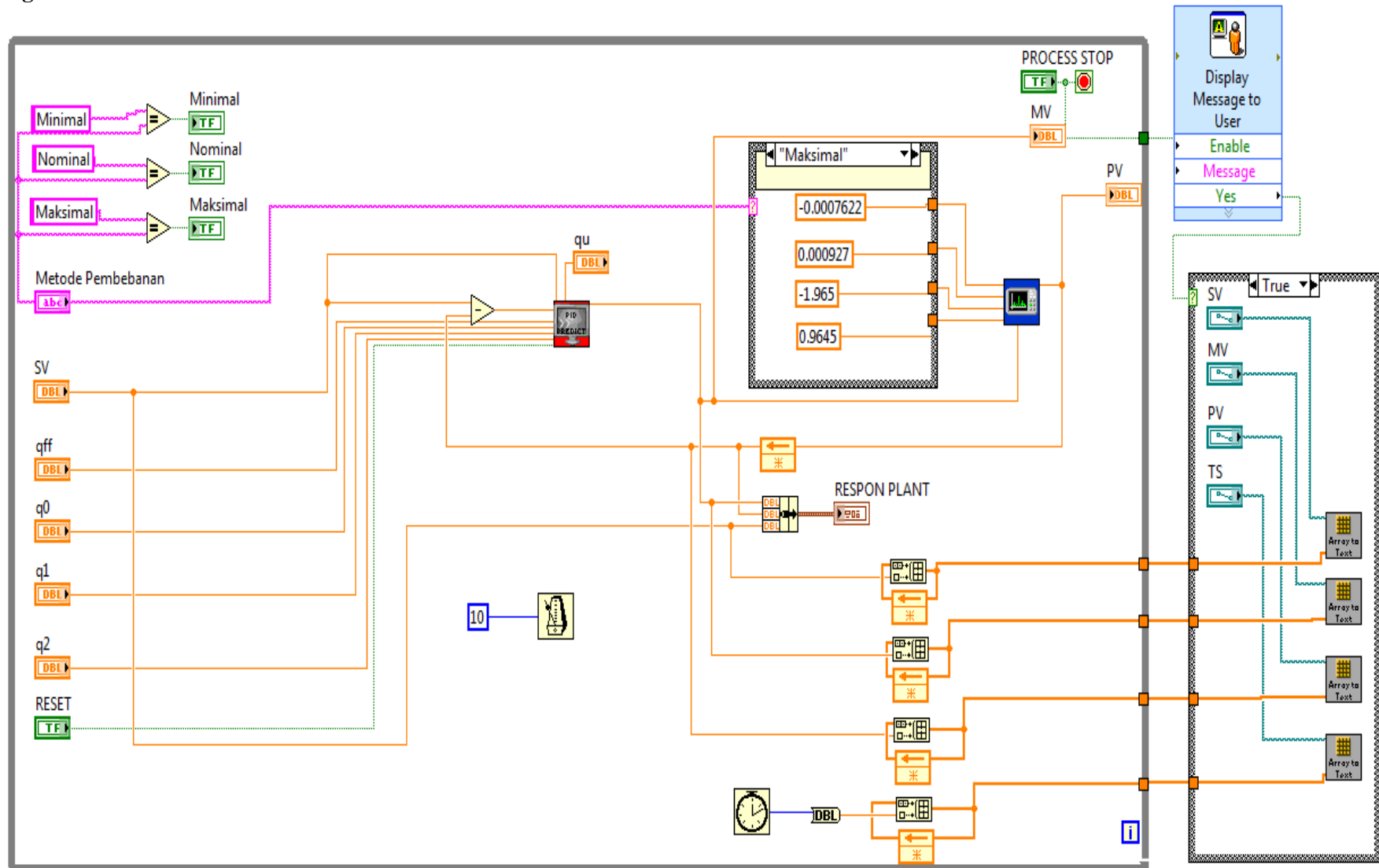
q1=sum(RecKPID(:,2));
disp('q1=')
disp(q1)
q2=sum(RecKPID(:,3));
disp('q2=')
disp(q2)

%% Mencari Gain PID Konvensional
kp=-q1-2*q2;
disp('kp=');
disp(kp)
ki=q0+q1+q2;
disp('ki=');
disp(ki)
kd=q2;
disp('kd=');
disp(kd)

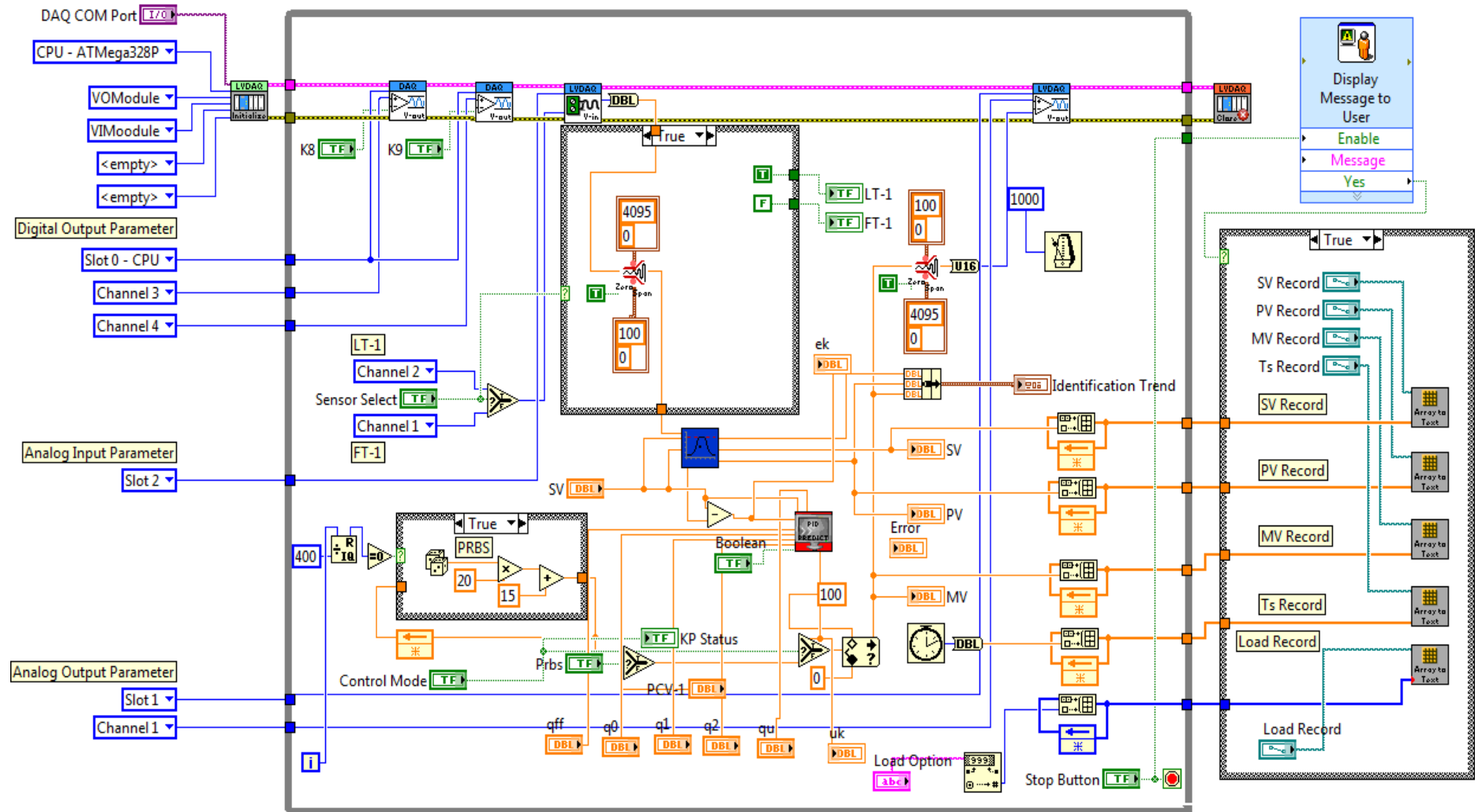
%% Mencari Gain FeedFoward
Kff=K*BarsR;
RecKff=Kff(1,:);
qff=sum(RecKff(:,1));
disp('qff=');
disp(qff)

```

E. Program Simulasi



F. Program Implementasi



RIWAYAT PENULIS



Adi Prasetyawan lahir di Lamongan pada tanggal 18 Januari 1993. Penulis telah menyelesaikan pendidikannya di SDN Gendong Kulon II, Lamongan, SMPN 1 Babat, dan SMAN 1 Babat. Pada tahun 2011, penulis melanjutkan studi di Institut Teknologi Sepuluh Nopember (ITS), Jurusan Teknik Elektro melalui jalur Seleksi Nasional Masuk Perguruan Tinggi Negeri (SNMPTN). Di Jurusan Teknik Elektro ini, penulis memilih Teknik Sistem Pengaturan sebagai bidang studi yang ditekuni. Pada bulan Juni 2015, penulis mengikuti seminar dan ujian Tugas Akhir sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Teknik Elektro.